Отчёт по лабораторной работе №5

Окорочкова Мария, M32341

17 декабря 2022 г.

Текущая конфигурация системы:

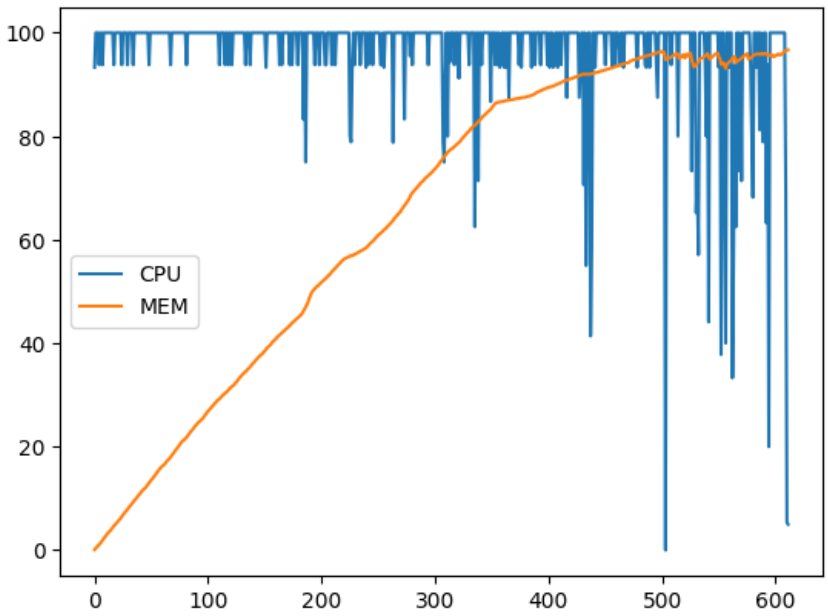
Общий объем оперативной памяти	12Gi
Объем раздела подкачки	4.0Gi
Объем свободной физической памяти в ненагруженной системе	10Gi
Объем свободного пространства в разделе подкачки в ненагруженной системе	2.5Gi
Размер страницы виртуальной памяти	4096B

Эксперимент №1

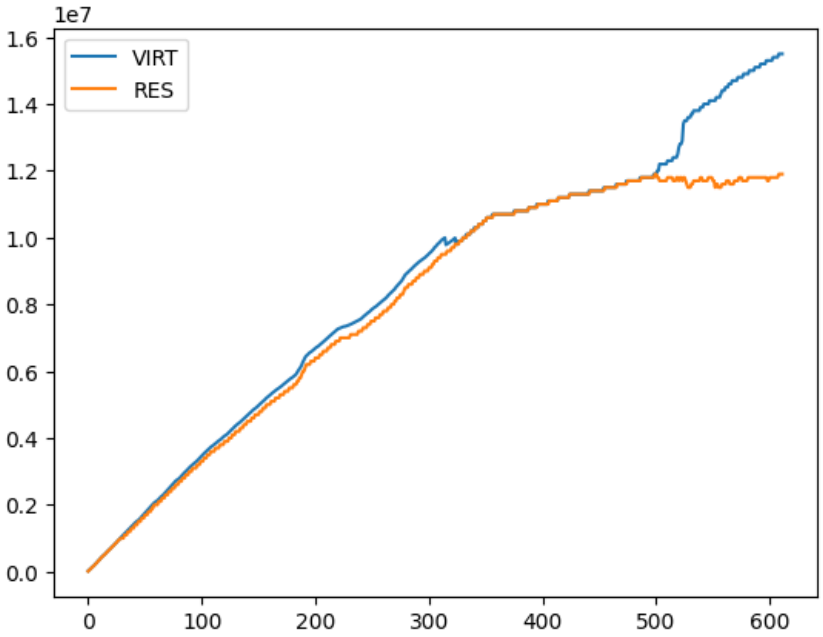
Первый этап

Значения параметров mem.bash

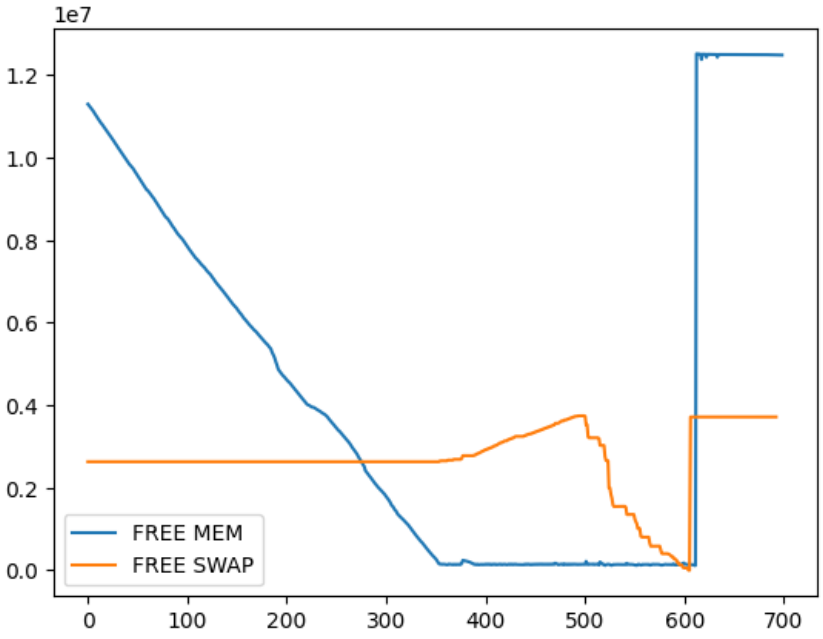
- PID, USER, PR, NI, SHR, S и COMMAND очевидно не изменялись (в следующих пунктктах учитываем это по умолчанию)
- TIME+ очевидно росло равномерно (в следующих пунктктах учитываем это по умолчанию)
- %MEM, %CPU:



- VIRT, RES:



- free MEM, free SWAP:



dmesg | grep "mem.bash":

```
[28872.410521] [  18073]  1000 18073  4064752  3118321 32620544   944342
0 mem.bash

[28872.410533] oom-kill:constraint=CONSTRAINT_NONE,nodemask=(null),cpuset=/,
mems_allowed=0,global_oom,task_memcg=/,task=mem.bash,pid=18073,uid=1000
```

```
[28872.411552] Out of memory: Killed process 18073 (mem.bash) total-vm:16259008kB,
anon-rss:12473284kB, file-rss:0kB, shmem-rss:0kB, UID:1000 pgtables:31856kB
oom_score_adj:0

[28873.006178] oom_reaper: reaped process 18073 (mem.bash), now anon-rss:0kB,
file-rss:0kB, shmem-rss:0kB
```

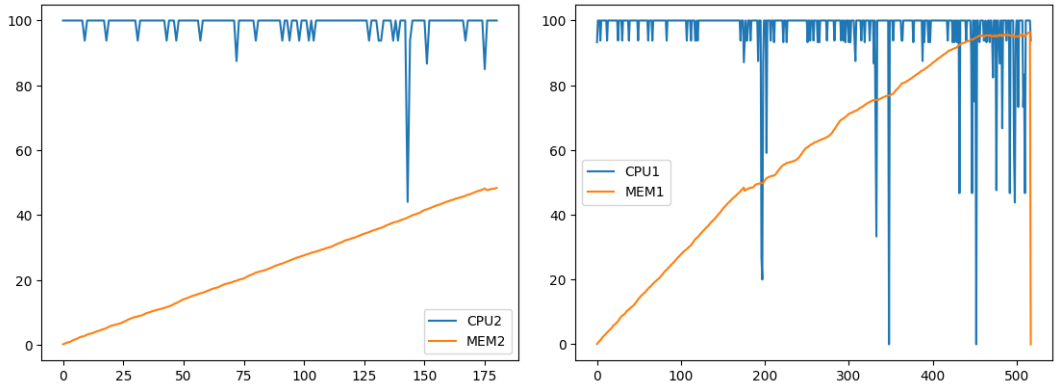
Значение последней строки в файле report.log — **208 000 000** — примерно максимальный размер массива. Это мы запомним для второго эксперимента!

Комментарий: Процесс убился, очевидно по всем графикам.

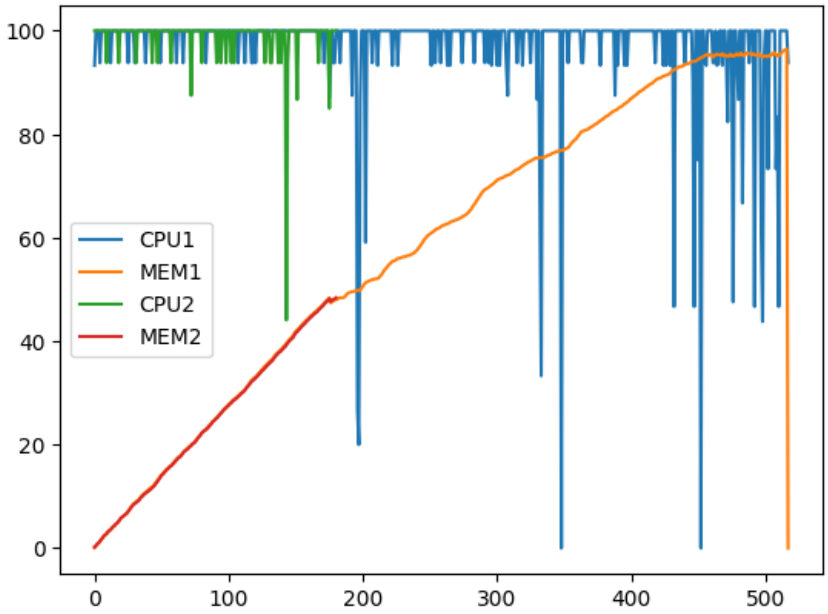
Второй этап

Значения параметров mem[2]*.bash

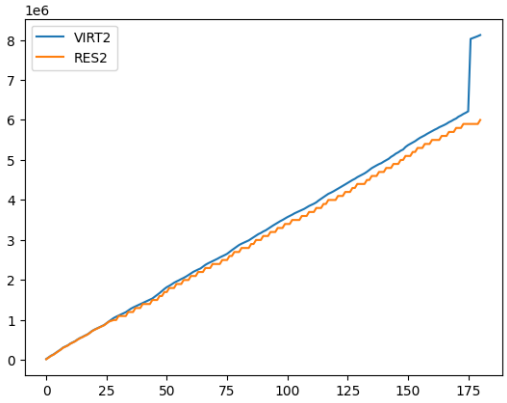
- mem.bash & mem2.bash: %MEM, %CPU:

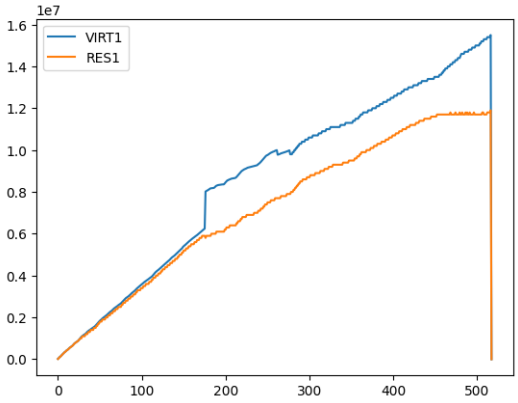


- mem.bash + mem2.bash: %MEM, %CPU:

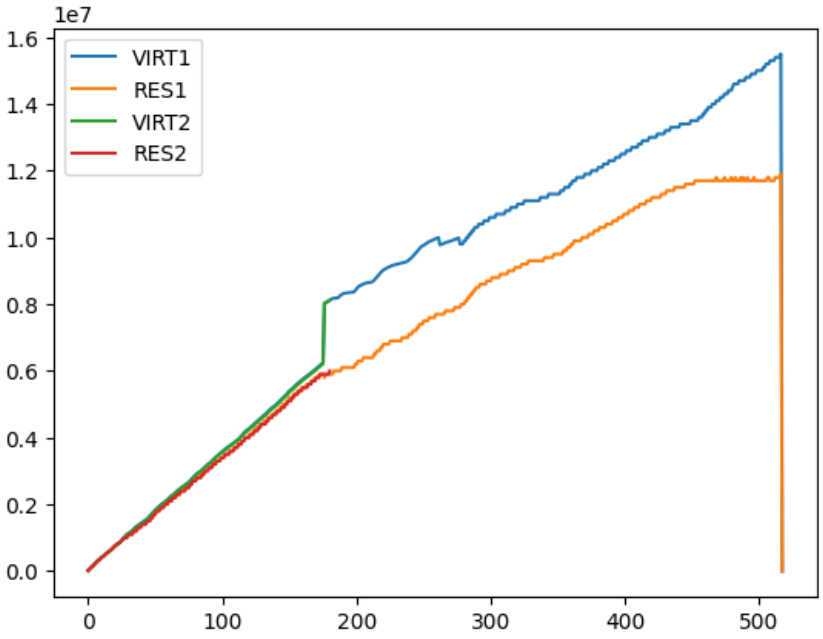


- mem.bash & mem2.bash: VIRT, RES:

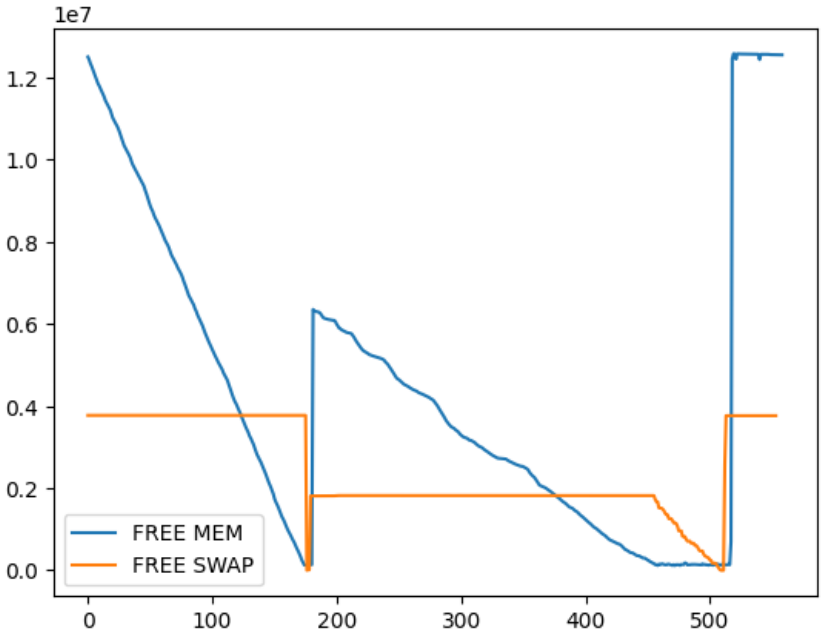




- mem.bash + mem2.bash: VIRT, RES:



- free MEM, free SWAP:



dmesg | grep "mem[2]*.bash":

```
[30613.250003] [    4884]  1000  4884  2032480  1556012 16334848   474371
0 mem.bash

[30613.250005] [    4885]  1000  4885  2032579  1563609 16322560   466869
0 mem2.bash

[30613.250039] oom-kill:constraint=CONSTRAINT_NONE,nodemask=(null),
cpuset=dns-forwarder,mems_allowed=0,global_oom,task_memcg=/,task=mem2.bash,
```

```
pid=4885,uid=1000

[30613.251011] Out of memory: Killed process 4885 (mem2.bash) total-vm:
8130316kB, anon-rss:6254436kB, file-rss:0kB, shmem-rss:0kB, UID:1000
pgtables:15940kB oom_score_adj:0

[30613.537030] oom_reaper: reaped process 4885 (mem2.bash), now anon-rss:
0kB, file-rss:0kB, shmem-rss:0kB

[30830.110434] [   4884]  1000  4884  4062376  3118018 32604160   942265
0 mem.bash

[30830.110703] oom-kill:constraint=CONSTRAINT_NONE,nodemask=(null),cpuset=/,
mems_allowed=0,global_oom ,task_memcg=/,task=mem.bash ,pid=4884,uid=1000

[30830.110729] Out of memory: Killed process 4884 (mem.bash) total-vm:
16249504kB, anon-rss:12472072kB, file-rss:0kB, shmem-rss:0kB, UID:1000
pgtables:31840kB oom_score_adj:0

[30830.784772] oom_reaper: reaped process 4884 (mem.bash), now anon-rss:0kB,
file-rss:0kB, shmem-rss:0kB
```

Значение последней строки в файлах:
report.log — **207 000 000** — примерно максимальный размер массива mem.bash; report2.log — **103 000 000** — примерно максимальный размер массива mem2.bash.

Комментарий: Показатели %MEM, VIRT, RES совпадают у mem2.bash и первой части mem.bash. Действительно, они и не могут быть различны. До момента, когда объём свободной физ. памяти и объём свободного пространства в разделе подкачки ушли в ноль, работали два запущенных процессов. Затем после переполнения аварийно завершился mem2.bash. Можно заметить, что после аварийно завершится и mem.bash.

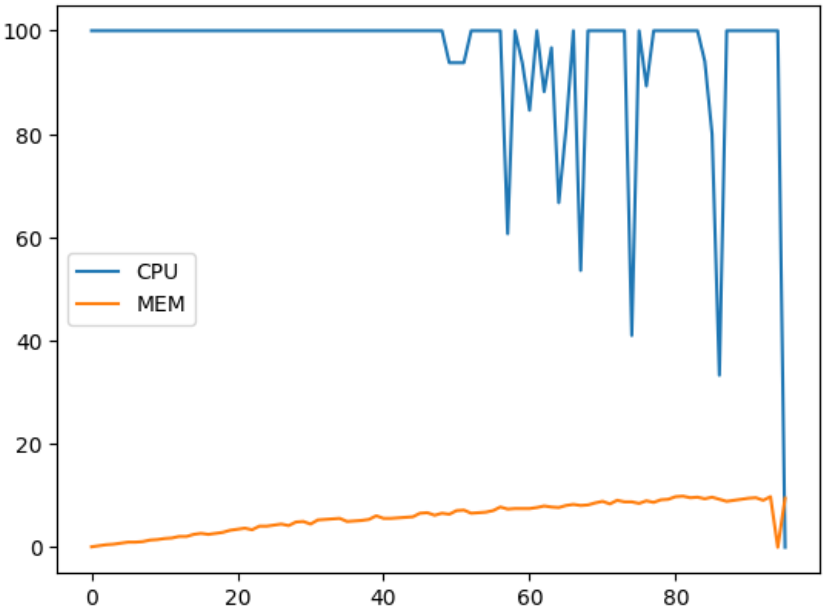
Итоги первого эксперимента:

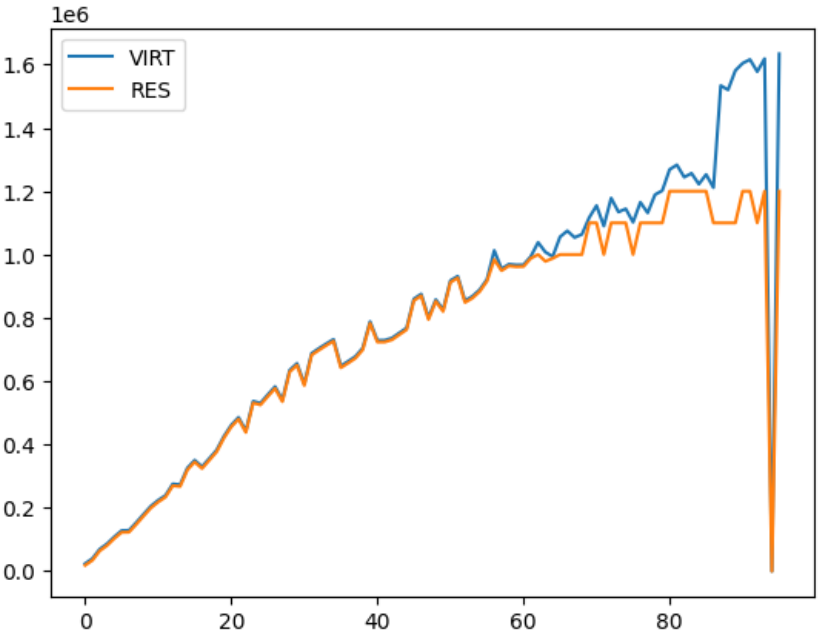
- %CPU процессов высоко, т.к. процесс ожидает только выделения дополнительной памяти. Это же объясняет неполную загрузку процессора процессами и падение загрузки перед аварийной остановкой.
- %MEM обозначает используемый объём физ. памяти, поэтому он сначала растёт. При использовании файла подкачки %MEM падает, т.к. часть страниц памяти, используемых процессом, переносится в swap. Это происходит скачкообразно, т.к. страницы переносятся не по одной. Аналогичное верно про RES.
- В эксперименте с двумя процессами %MEM и RES первого mem.bash процесса не уменьшаются перед остановкой процесса, т.к. почти весь swap уже занят.
- В случае запуска одного скрипта количество элементов массива при аварийной остановке $\approx 2.08 \cdot 10^8$, т.к. $1.3 \cdot 10^9$ (приблизительно объем используемой памяти в байтах) $/ 2.08 \cdot 10^8 \approx 6.25$ байт. bash использует числовые переменные размером 64 бит.
- При запуске двух скриптов второй скрипт останавливается при использовании примерно половины от $2.07 \cdot 10^8$ элементов, т.к. первый скрипт использует столько же и в сумме используется вся доступная память. После остановки второго скрипта первый останавливается при $2.07 \cdot 10^8$ элементов, т.к. теперь память использует только он и фоновые процессы.

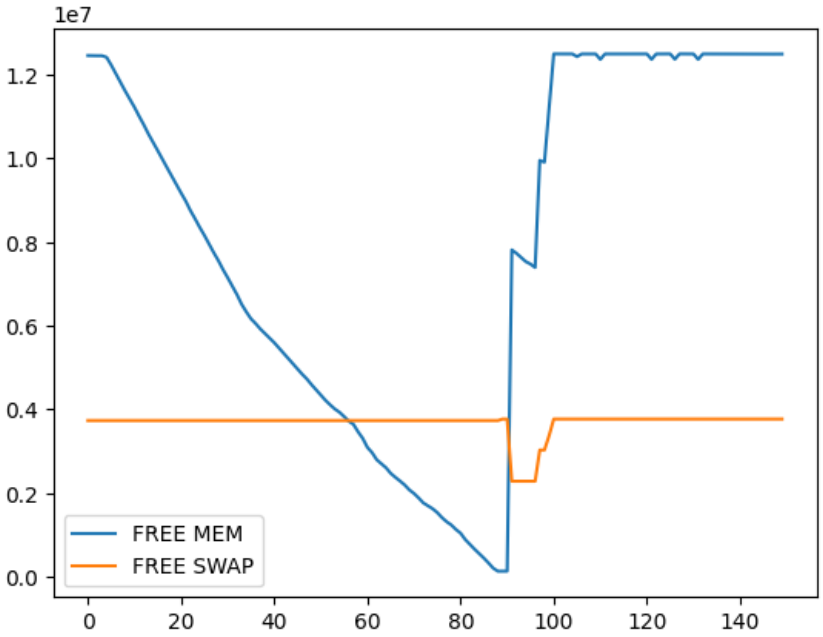
Выводы: Ура, провели эксперименты, построили таблички! Теор. знания подкреплены анализом эксперимента.

Эксперимент №2

Рассмотрим N = 20 800 000 K = 10

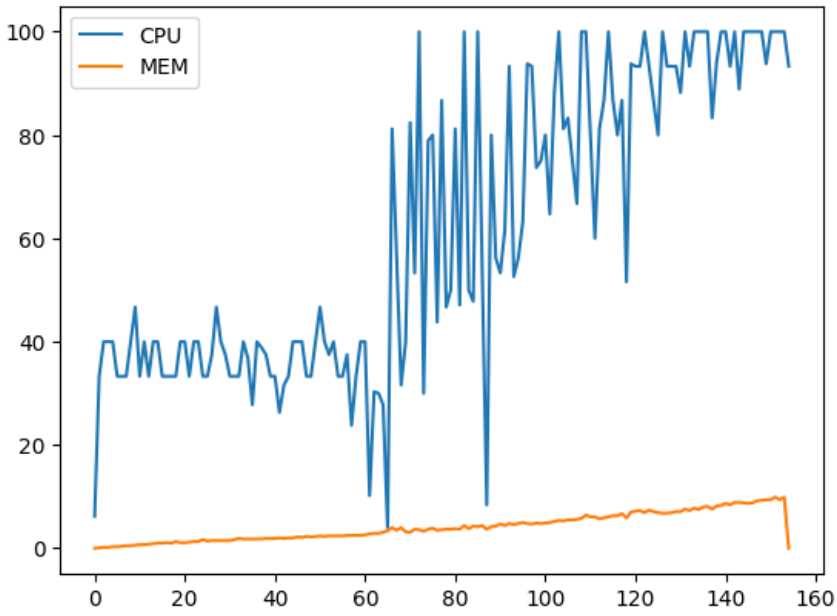
- 

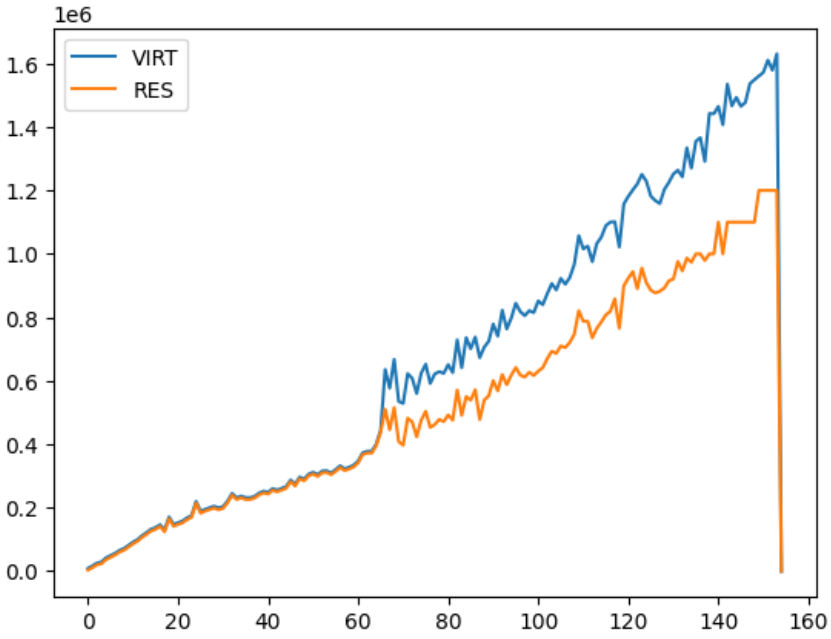
- 

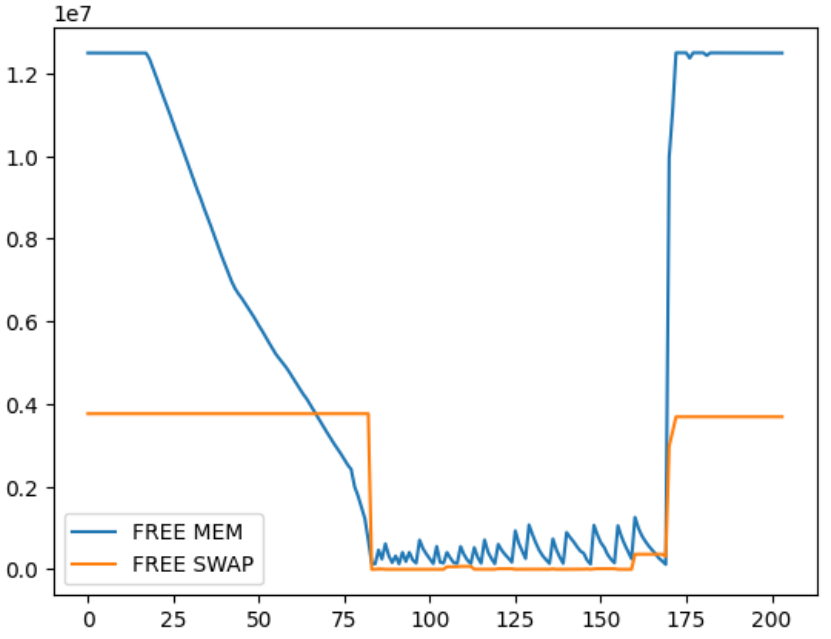


Комментарий: Процесс аварийно не завершился, журнал пуст, весьма логично: в первом эксперименте размер массива как раз и был 20 800 000 * 10. (Да, free mem обнулился, но использование swap продолжилось недолго и этого хватило.)

Рассмотрим N = 20 800 000 K = 30



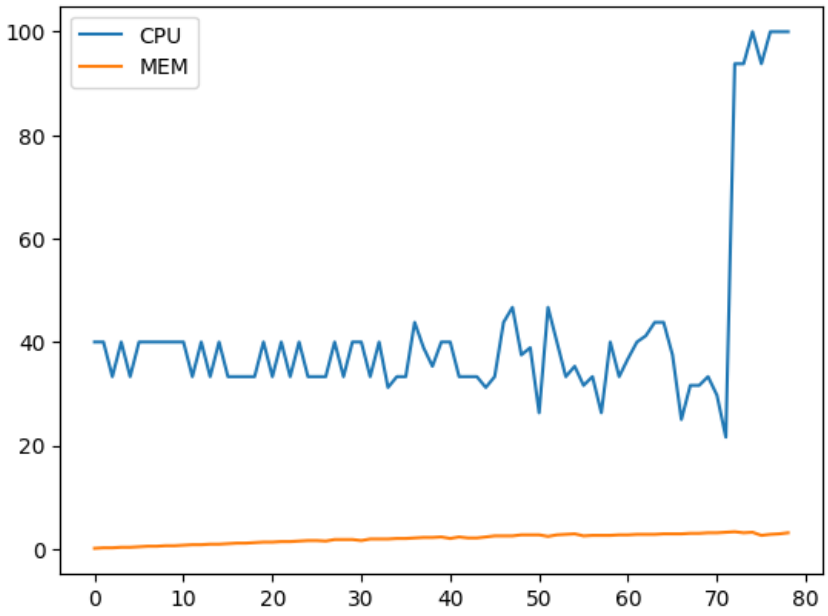


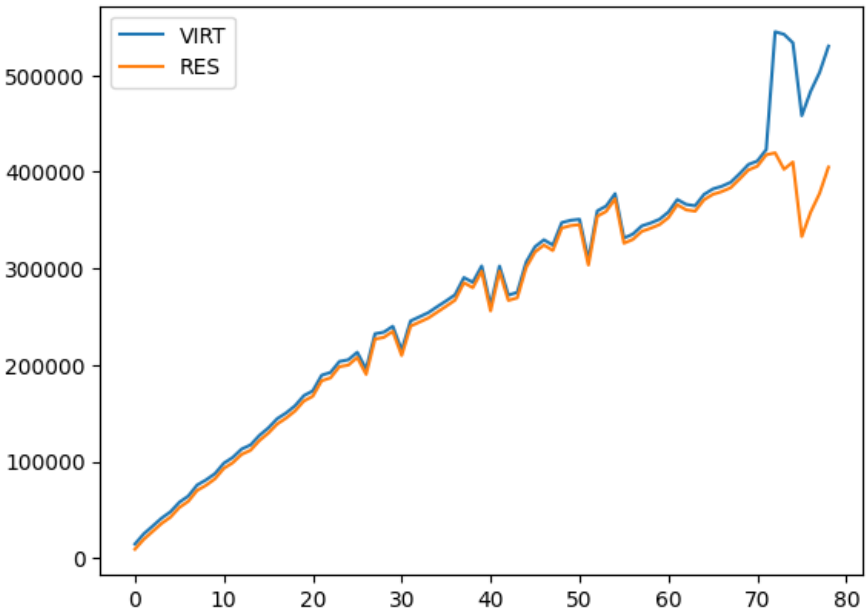


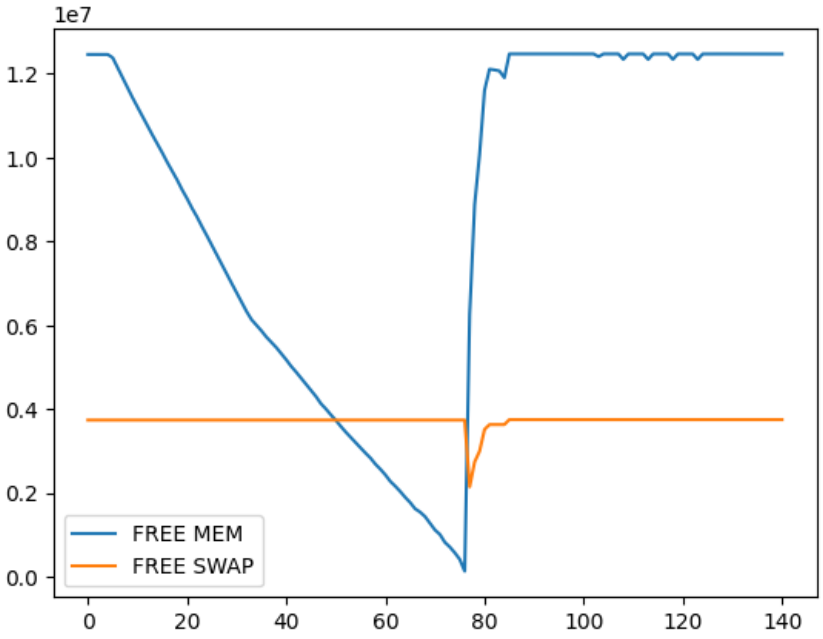
Результат **dmesg | grep "newmem.bash"** лежит в файле dmesg1.log.

Комментарий: Процесс аварийно завершился. Потому что 20 800 000 * 30 > 20 800 000 * 10.

Рассмотрим N = 7 000 000 K = 30

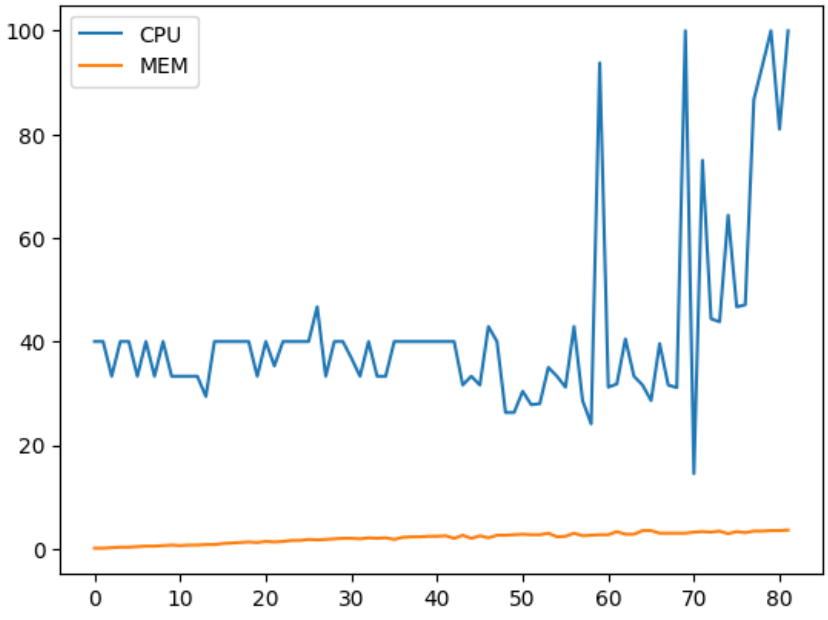
- 

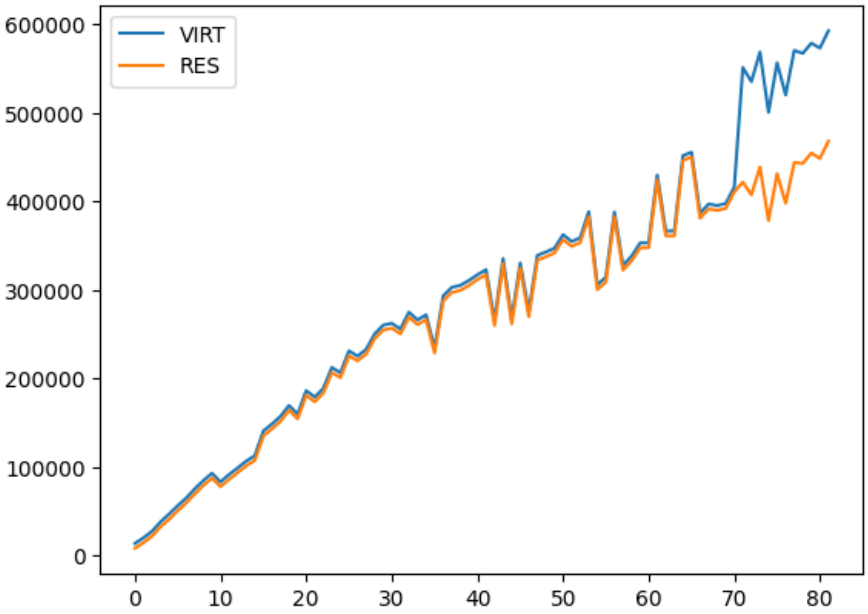
- 

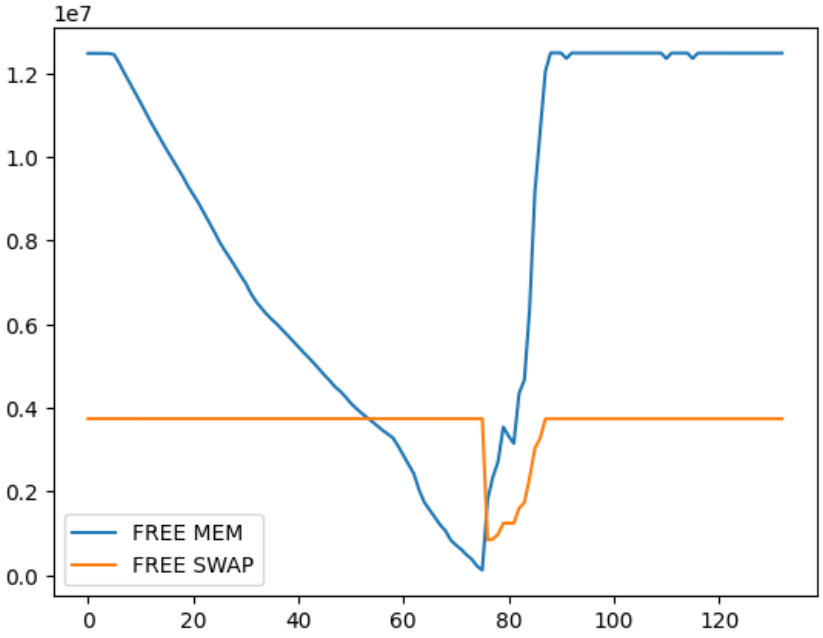


Комментарий: Мы взяли $N = 7\ 000\ 000$, потому что $208\ 000\ 000 / 30 = 6\ 933\ 333.(3) \approx 7\ 000\ 000$. Процесс не завершился аварийно, журнал пуст.

Рассмотрим N = 7 500 000 K = 30

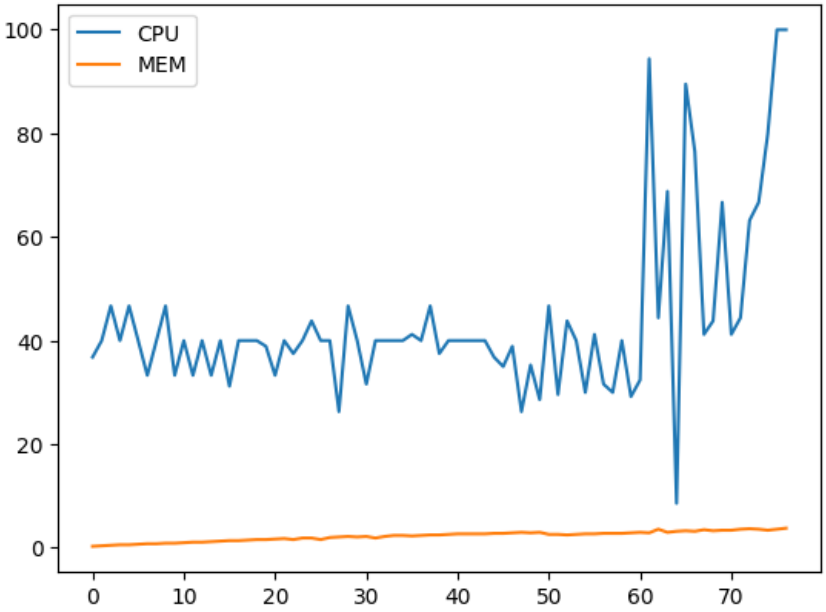


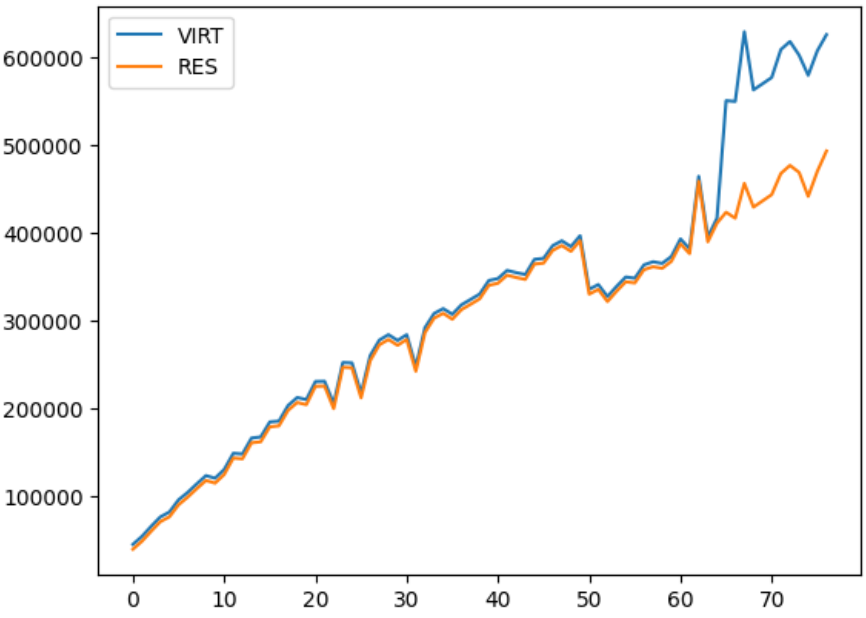
- 

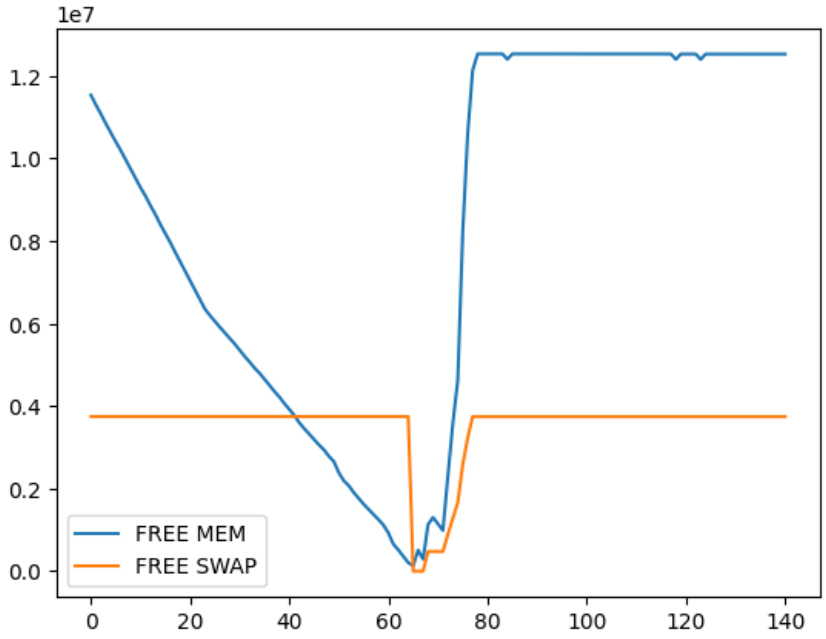
- 

Комментарий: Процесс не завершился аварийно, НО видно, что free swap стремится вниз. Максимальное значение N, чтобы при K=30 не происходило аварийных завершений процессов, существует при free swap, находящемся на ε расстоянии от нуля.

Рассмотрим N = 8 000 000 K = 30

- 

- 



Результат **dmesg | grep "newmem.bash"** лежит в файле dmesg2.log.
Комментарий: Процесс аварийно завершился.

Итоги второго эксперимента:

- При N = 20 800 000, K = 10 всё круто.
- При N = 20 800 000, K = 30 ряд процессов закончился аварийно, т.к. процессы требуют суммарно приблизительно в 3 раза больше оперативной памяти, чем доступно.
- Максимальное значение N, такое что при K=30 не происходило аварийных остановок процессов $\approx 7.5 \cdot 10^6$ (если быть точнее: оно между 7.5 и 8). Оно отличается от ожидаемого значения — $2.08 \cdot 10^8/30 \approx 7 \cdot 10^6$, т.к. процессы не работают синхронно и некоторые достигают штатного завершения раньше других, пока оперативной памяти хватает.