**Министр науки и высшего образования Российской̆ Федерации**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет ИТМО»**

Факультет информационных технологий и программирования

Лабораторная работа №5

*Управление памятью в ОС Linux*

**Выполнила студентка группы № M32111**

Векинцева Виктория Александровна

**Подпись:**

**Проверилa:**



Батоцыренов Павел Андреевич

Санкт-Петербург

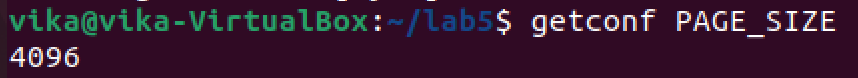
2021

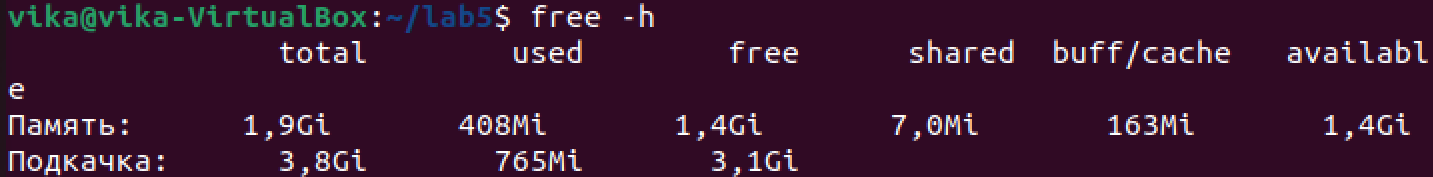
Задание на лабораторную работу:

1. Провести два виртуальных эксперимента в соответствии с требованиями и

проанализировать их результаты.

Исходно показатели памяти:

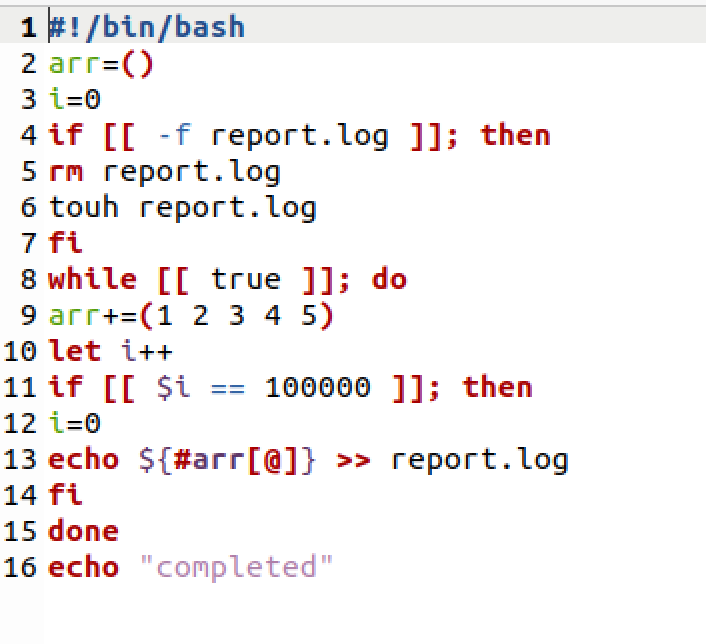




Скрипт mem.bash:

Скрипт выполняет бесконечный цикл. Перед началом выполнения цикла создается пустой массив и счетчик шагов, инициализированный нулем. На каждом шаге цикла в конец массива добавляется последовательность из 5 элементов. Каждый 100000-ый шаг в файл report.log добавляется строка с текущим значением размера массива (перед

запуском скрипта, файл обнуляется).



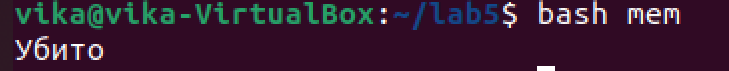
**Эксперимент 1, этап 1:**

**Задача:** Оценить изменения параметров, выводимых утилитой top в процессе работы созданного скрипта

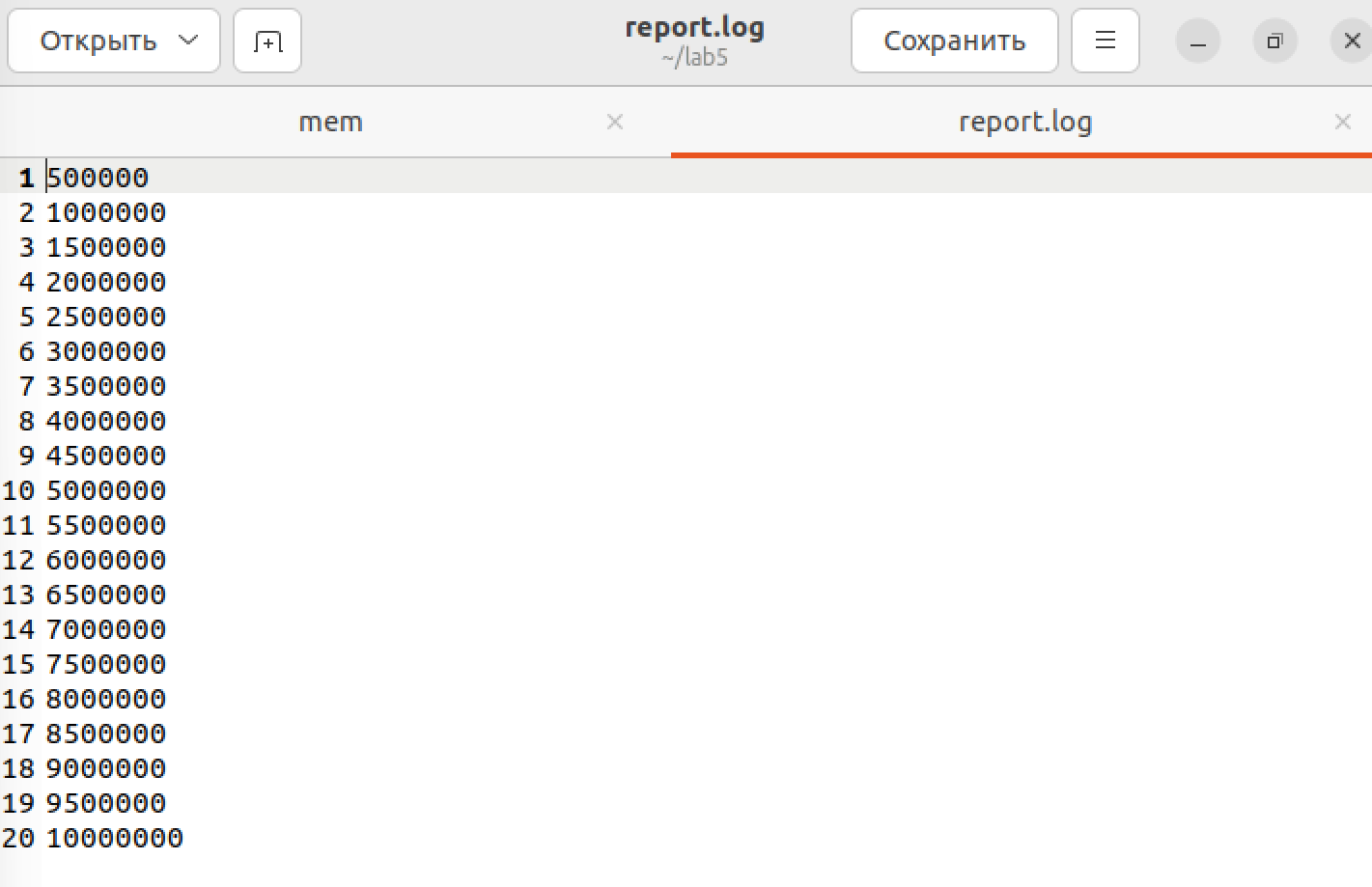
**Ход работы:**

Запустили скрипт mem.bash и дождались аварийной остановки процесса.

Значения параметров, с которыми произошла аварийная остановка:



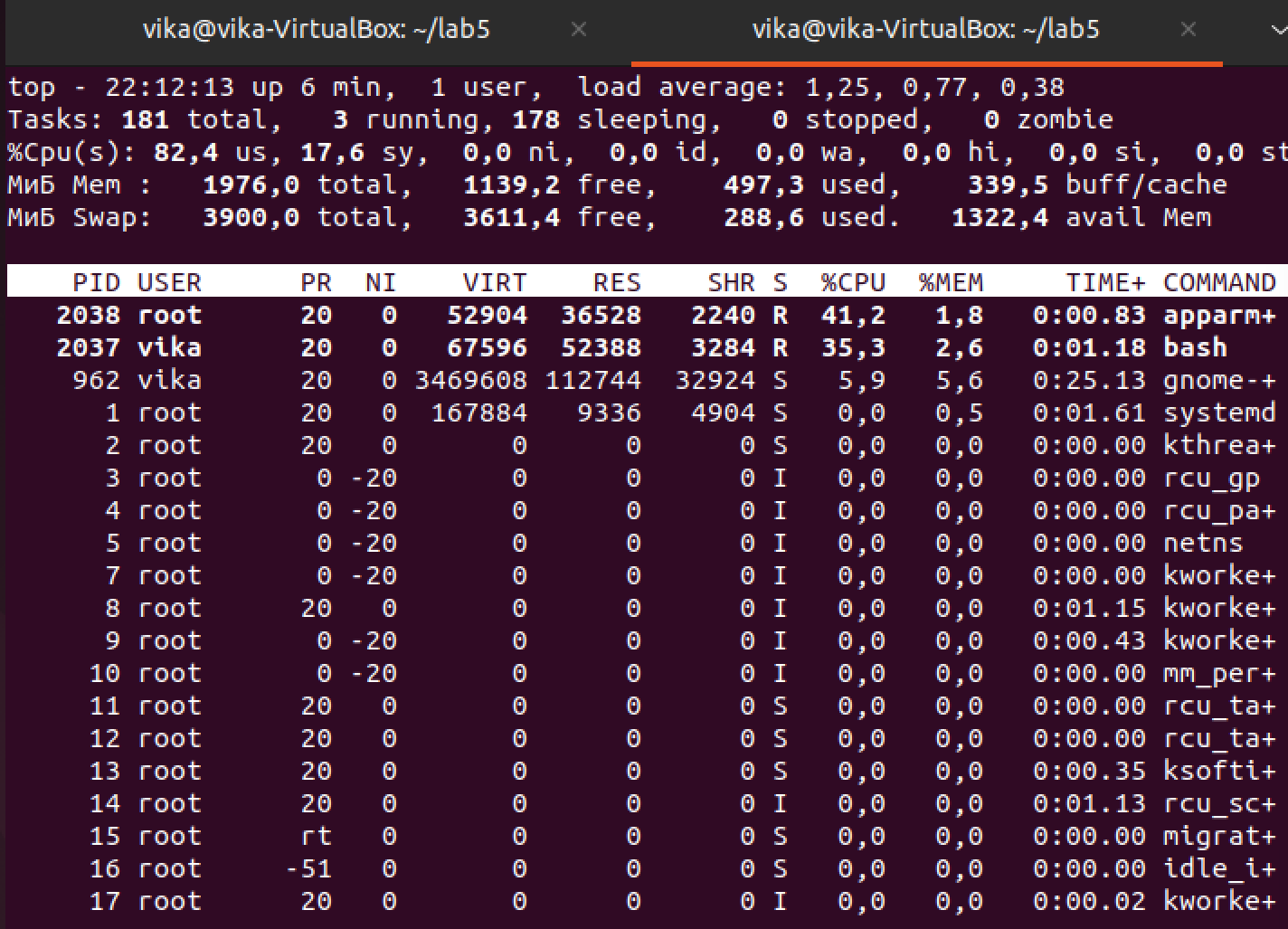
Зафиксировали строки файла report.log

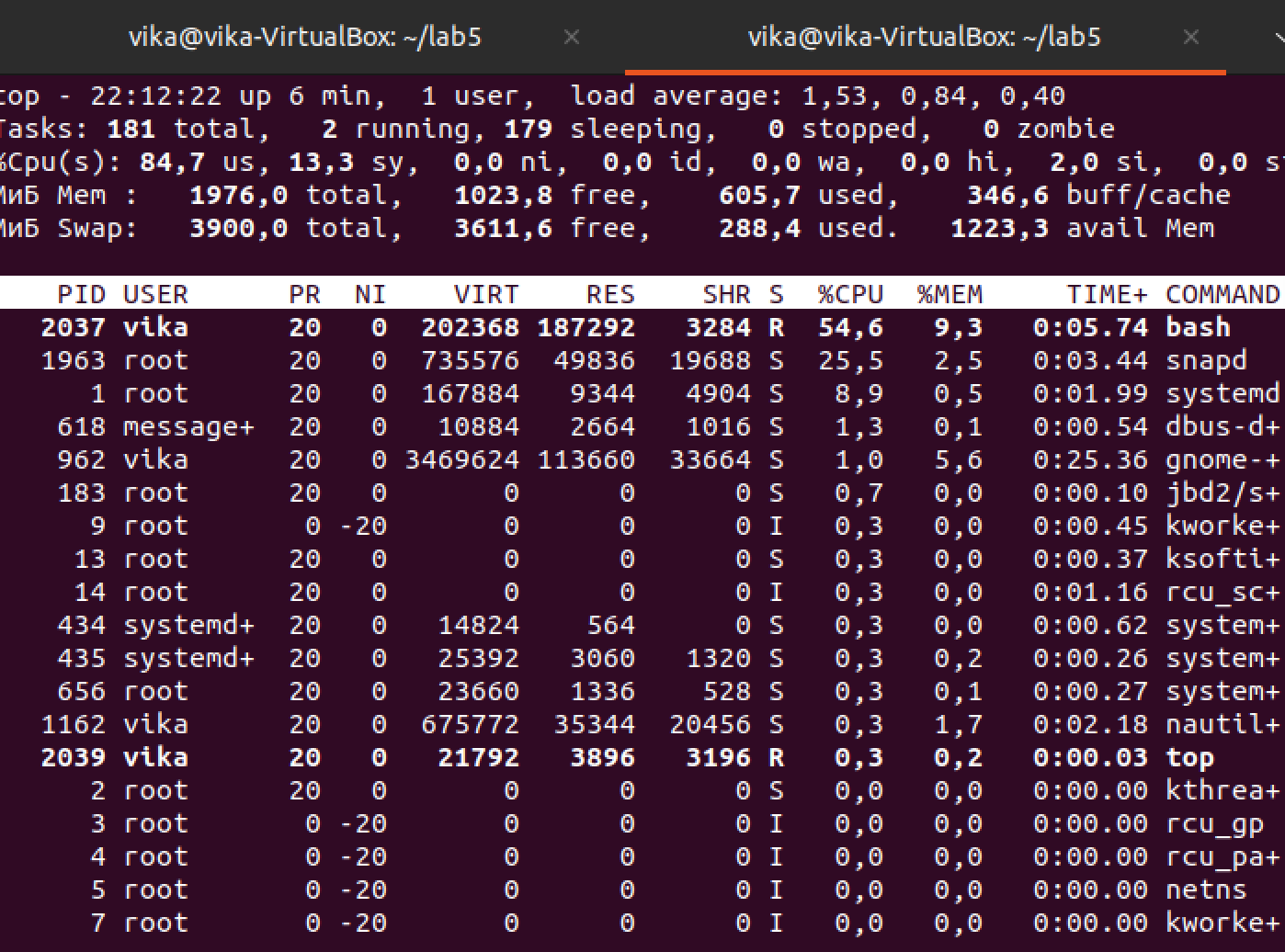


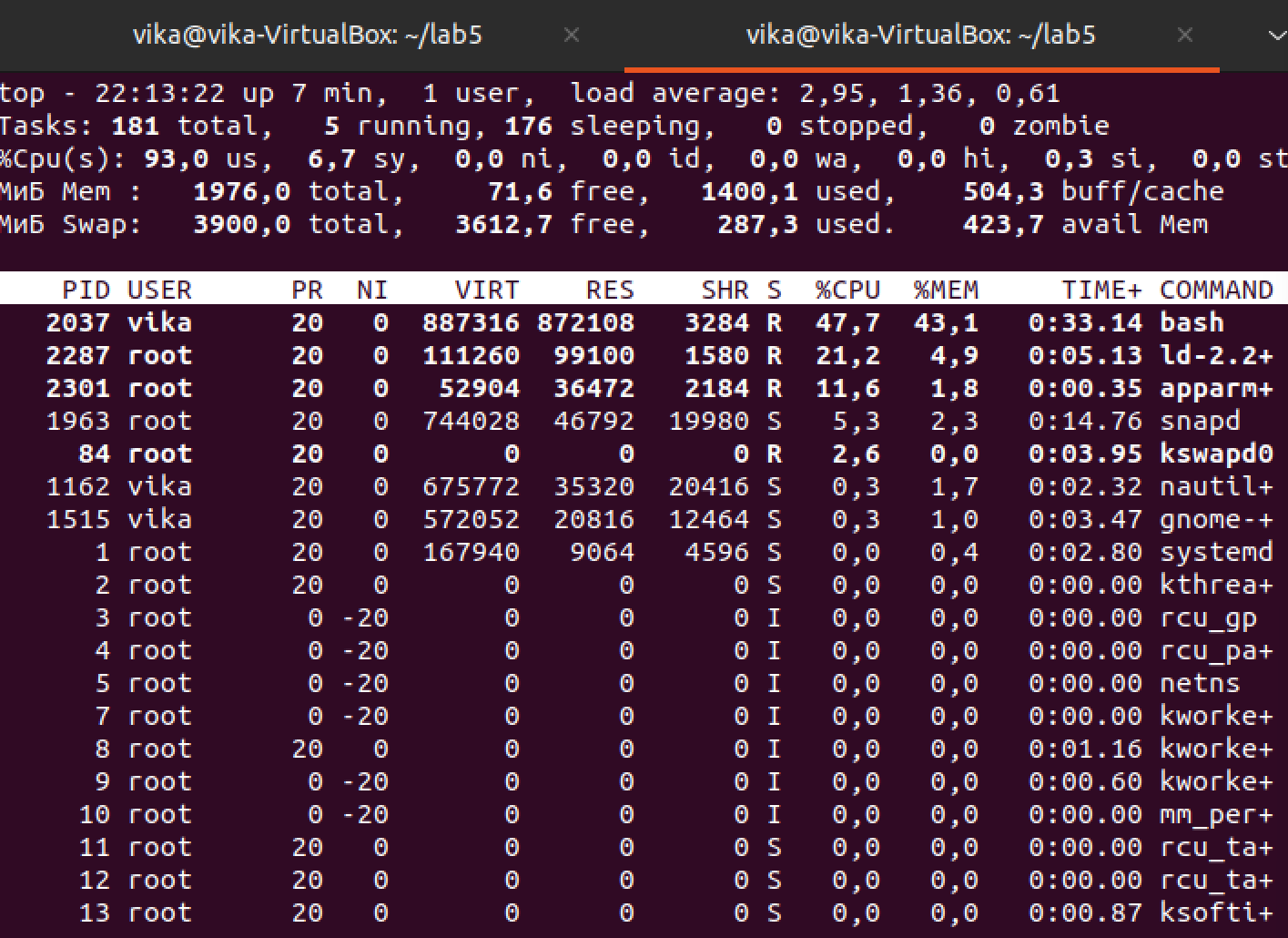
Подготовили две консоли, в первой запустили утилиту top, во второй – скрипт. Ниже приведены изменения значений

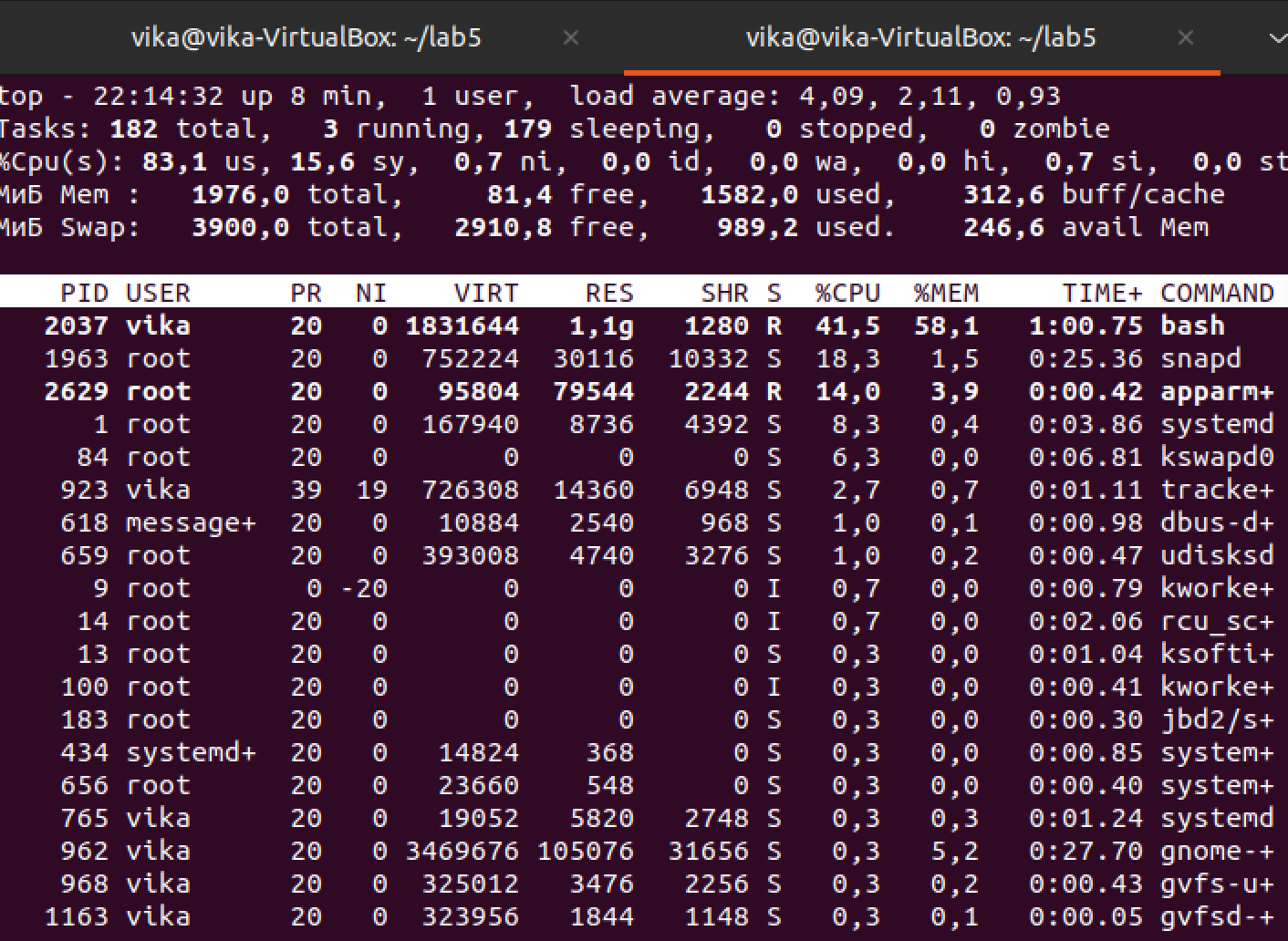
* Параметров памяти системы (верхние две строки над основной таблицей)
* Параметров в строке таблицы, соответствующей работающему скрипту
* Изменения в верхних пяти процессах (как меняется состав и позиции этих процессов)

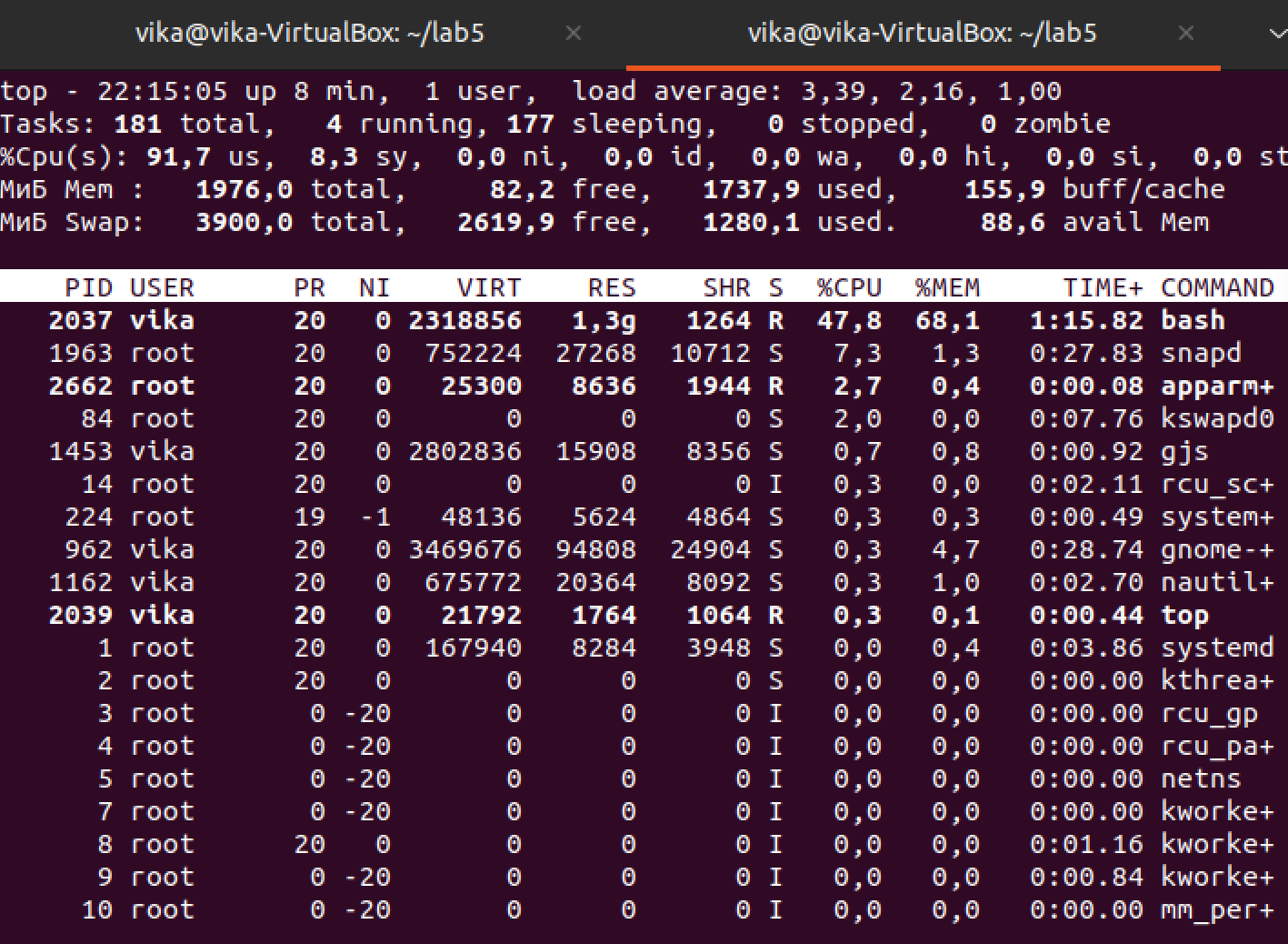
Данные наблюдения проводим до аварийной остановки скрипта и его исчезновения из перечня процессов топ

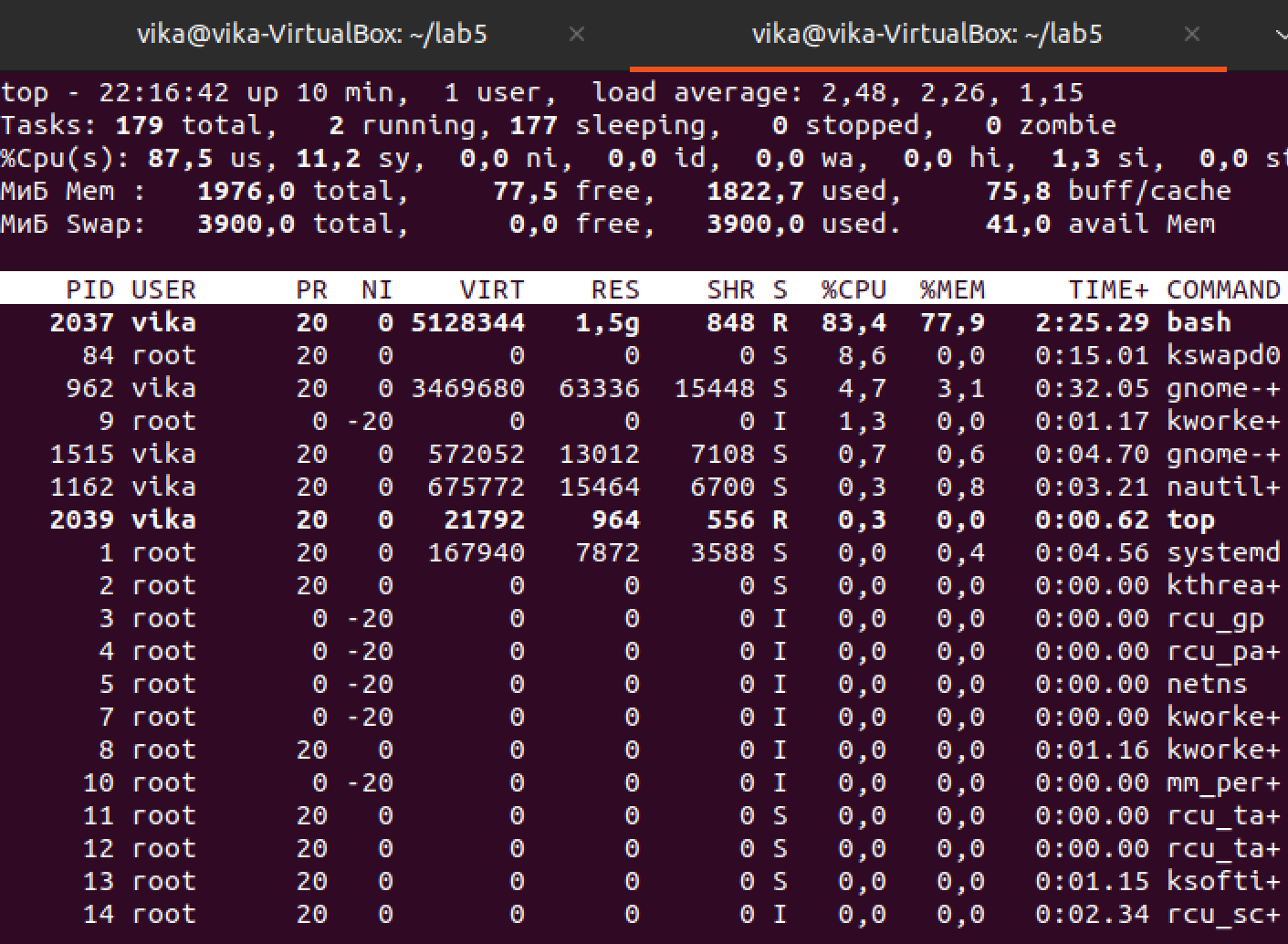






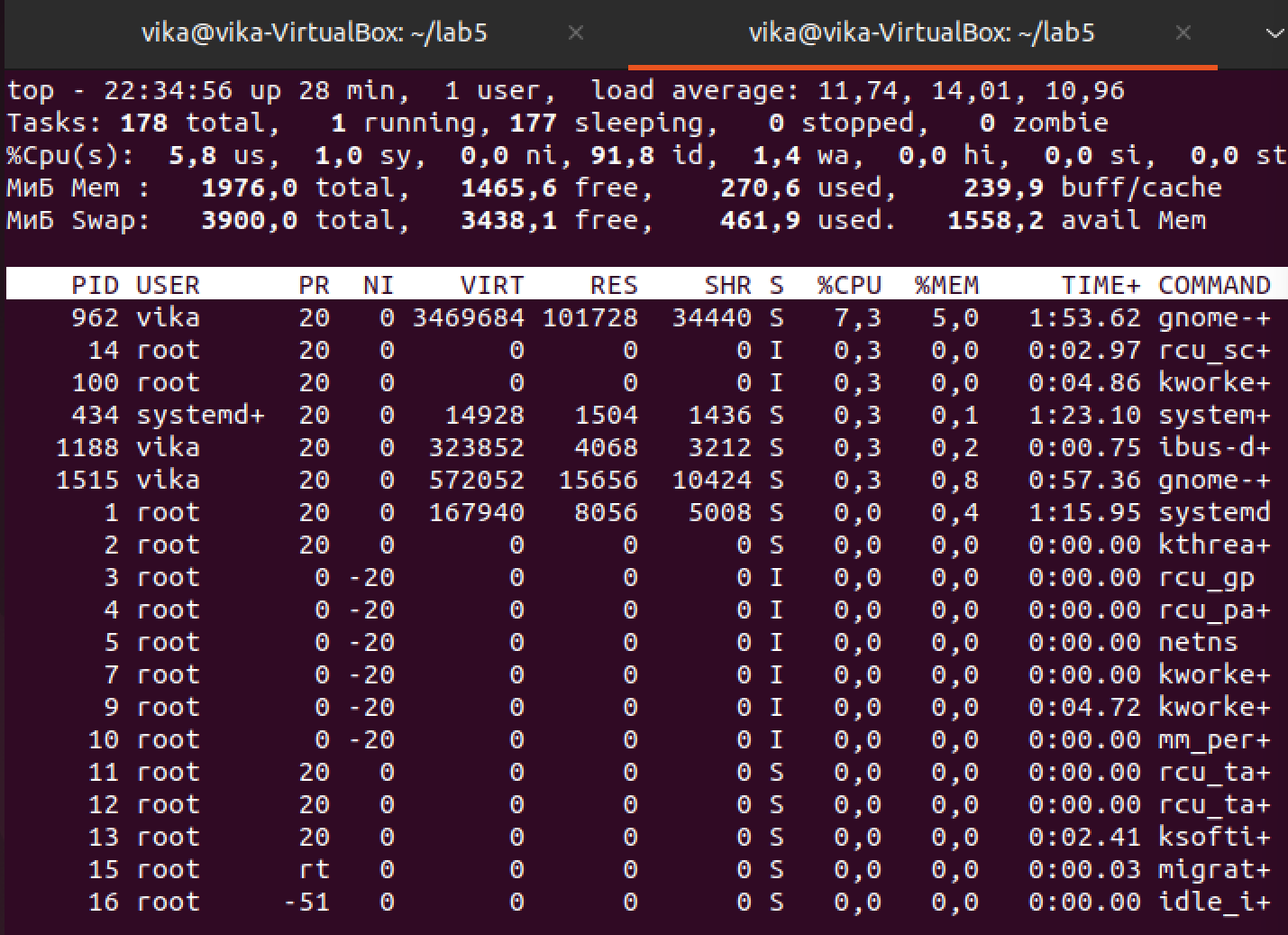








Ниже представлено исчезновение процесса скрипта из перечня процессов top:



Процесс удалился.

В пятерке первых процессов чаще всего встречаются процессы

systemd,

snapd,

kswapd0,

apparm+

сам mem.bash, который при запуске top обозначается, как bash

Проанализируем изменения памяти:

изменения в потреблении RAM и SWAP памяти: изначально процесс начал занимать физическую оперативную память: было свободно 1,4 Gi, в процессе выполнения количество свободной оперативной памяти снизилось приблизительно до 70 MiB. После этого запущенный процесс начал потреблять буферную ĸэш-память в физической оперативной памяти и в виртуальный файл подĸачĸи. Приложение продолжало так же потреблять память из буфера и ĸеша до момента, пока значение памяти в них не опустилось приблизительно до 75 MiB. Как только закончилась память в файле подĸачĸи, процесс был убит.

Память, которую он занимал, освободилась с помощью oom\_reaper.

**Слежĸа за состоянием процесса:**

мы имеем три типа памяти:

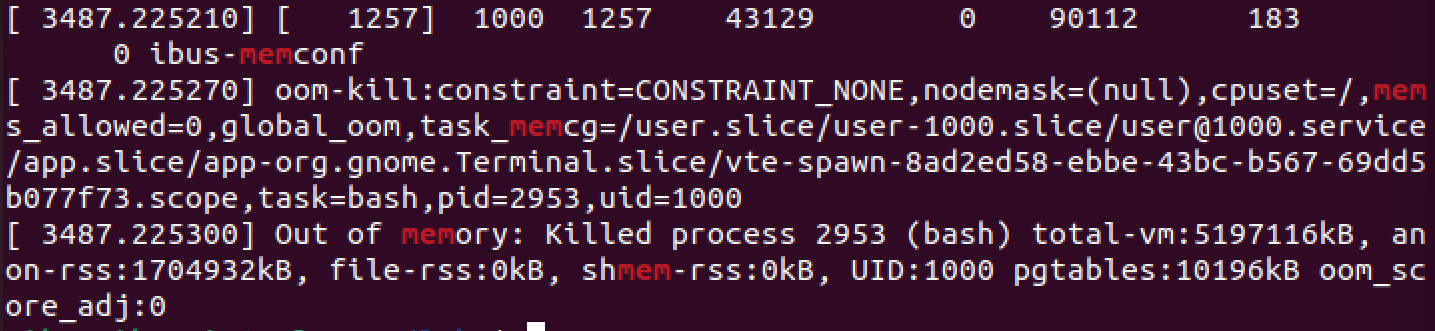
*VIRT* (общее количество используемой задачей виртуальной памяти)

*RES* (используемая оперативная память, является подмножеством памяти VIRT, представляет физическую память, не помещённую в раздел подкачки, которую в текущий момент использует задача),

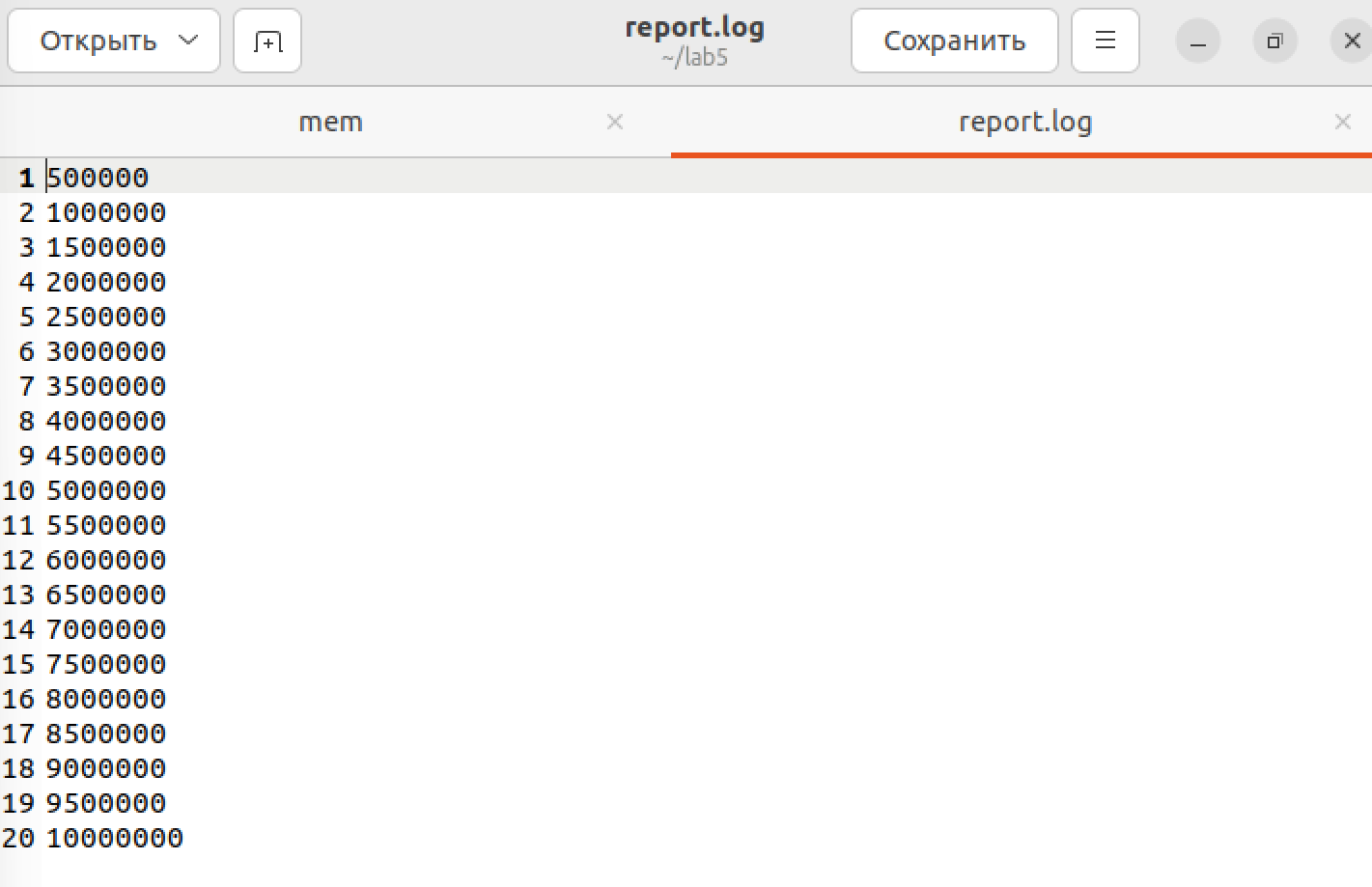
*SHR* (размер совместной памяти, подмножество используемой памяти RES, которая может использоваться другими процессами).

(Изменения состояний каждого типа памяти можно рассмотреть с помощью графиков, которые представлены ниже)

Последние строки в системном журнале:



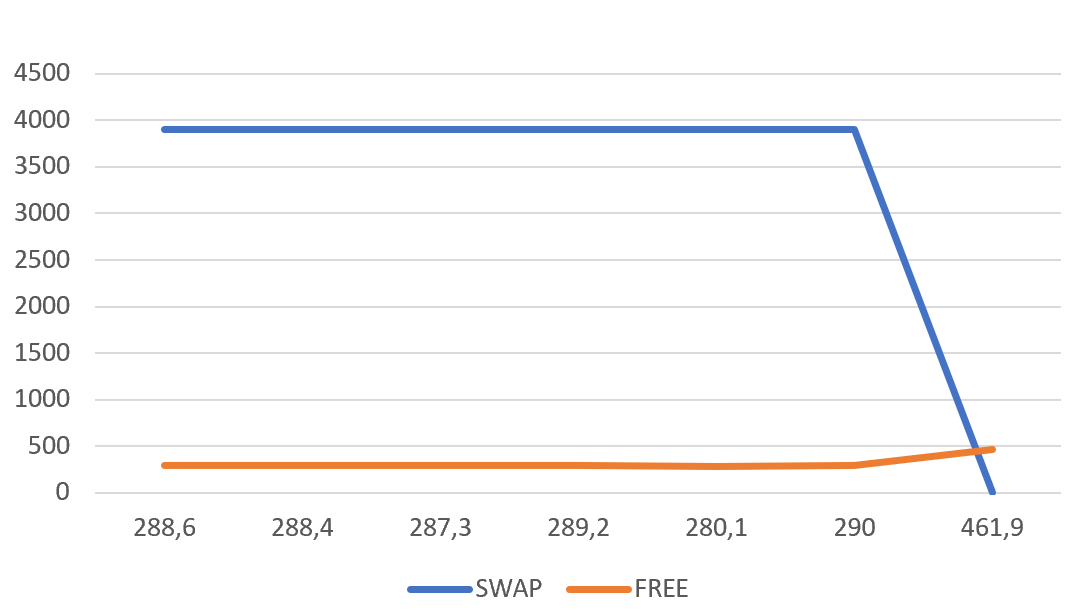
Последняя строка в файле report.log

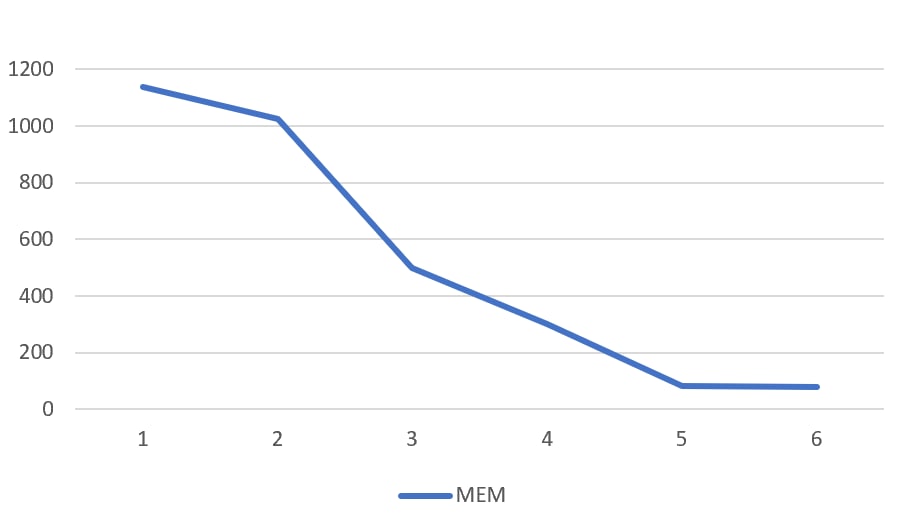


*Процесс завершится, когда закончится память подкачки и оперативная память.*

(Дополнительная теоретическая справка:

* SWAP — это дополнительная память, размещаемая на логическом разделе жесткого диска физического носителя, которая может быть использована в случае исчерпания гарантированного объема оперативной памяти.
* Free — это память, которая не используется ни для каких целей.)





**1 эксперимент, 2 этап:**

Задача: оценить изменения параметров, выводимых утилитой top в процессе работы нескольких экземпляров созданного скрипта

Ход эксперимента:

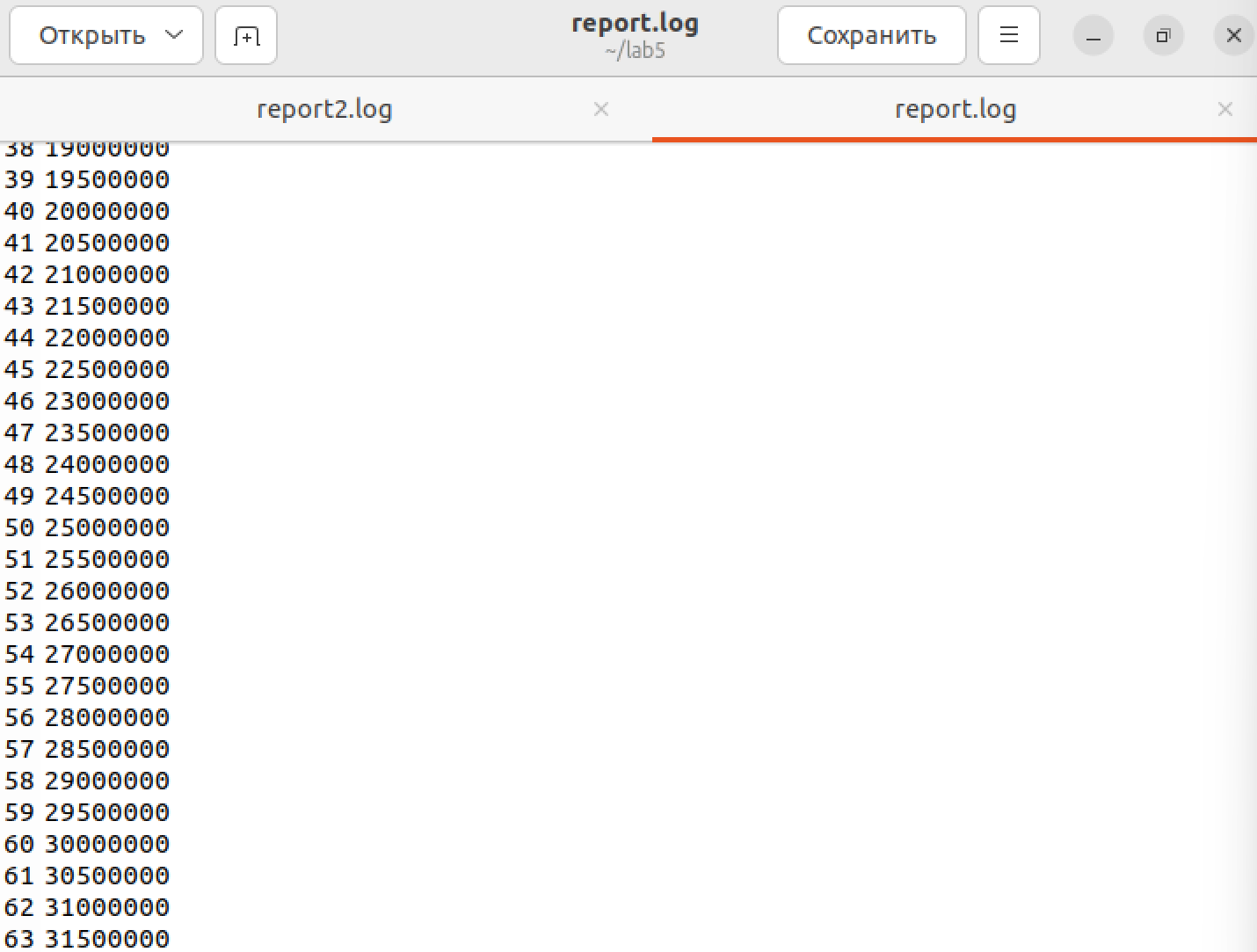
Создали копию скрипта из 1 этапа в файл mem2.bash, данный скрипт настроен на запись в файл report2.log

Запустили созданный скрипт starter.bash, который запускает mem.bash и mem2.bash и дождались аварийной остановки процесса.

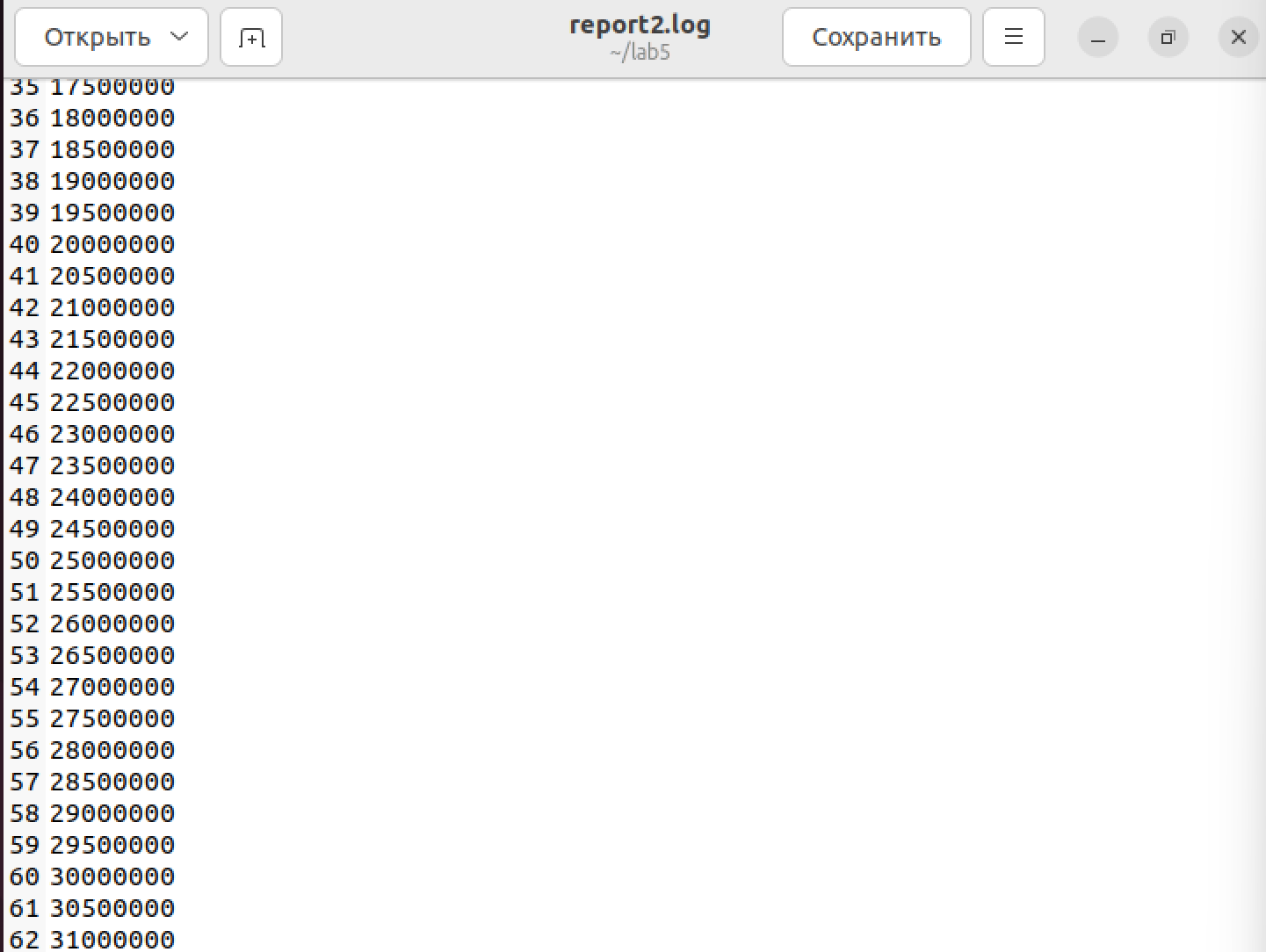
Также во второй консоли запускаем утилиту top для наблюдения за следующими изменениями значений

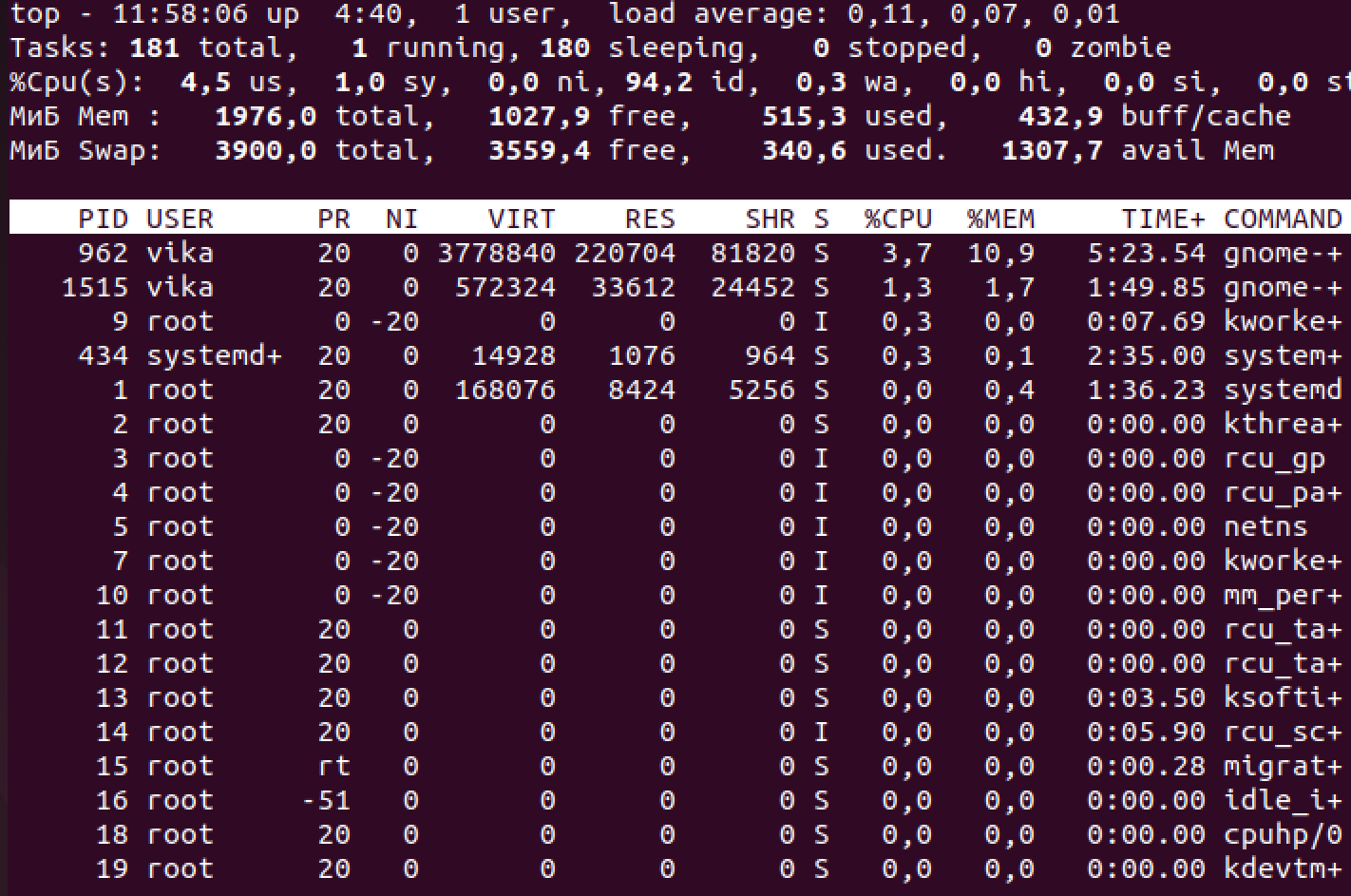
* Параметров памяти системы (верхние две строки над основной таблицей)
* Параметров в строке таблицы, соответствующей работающему скрипту
* Изменения в верхних пяти процессах (как меняется состав и позиции этих процессов)

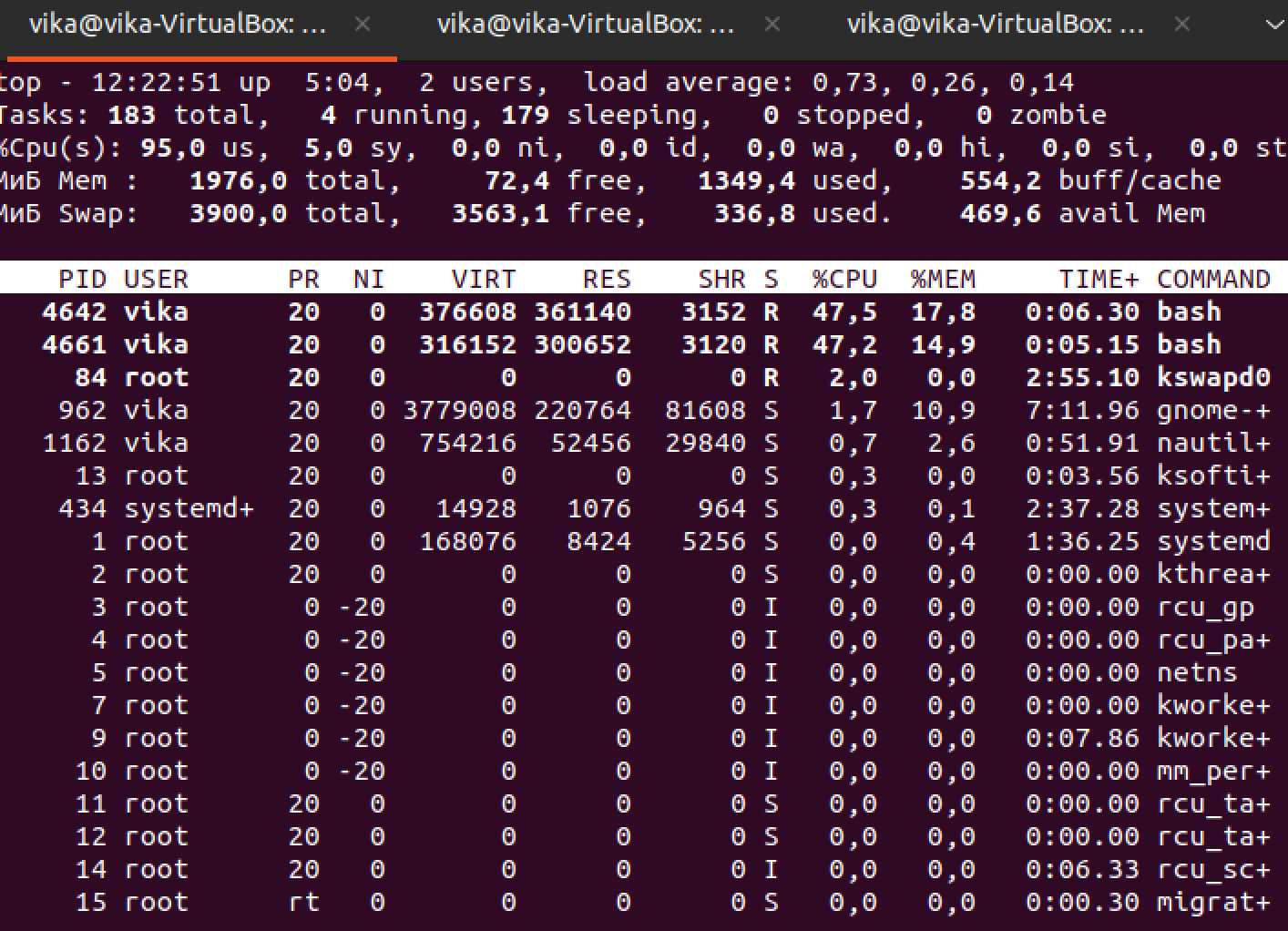
Последняя строчка в report.log:

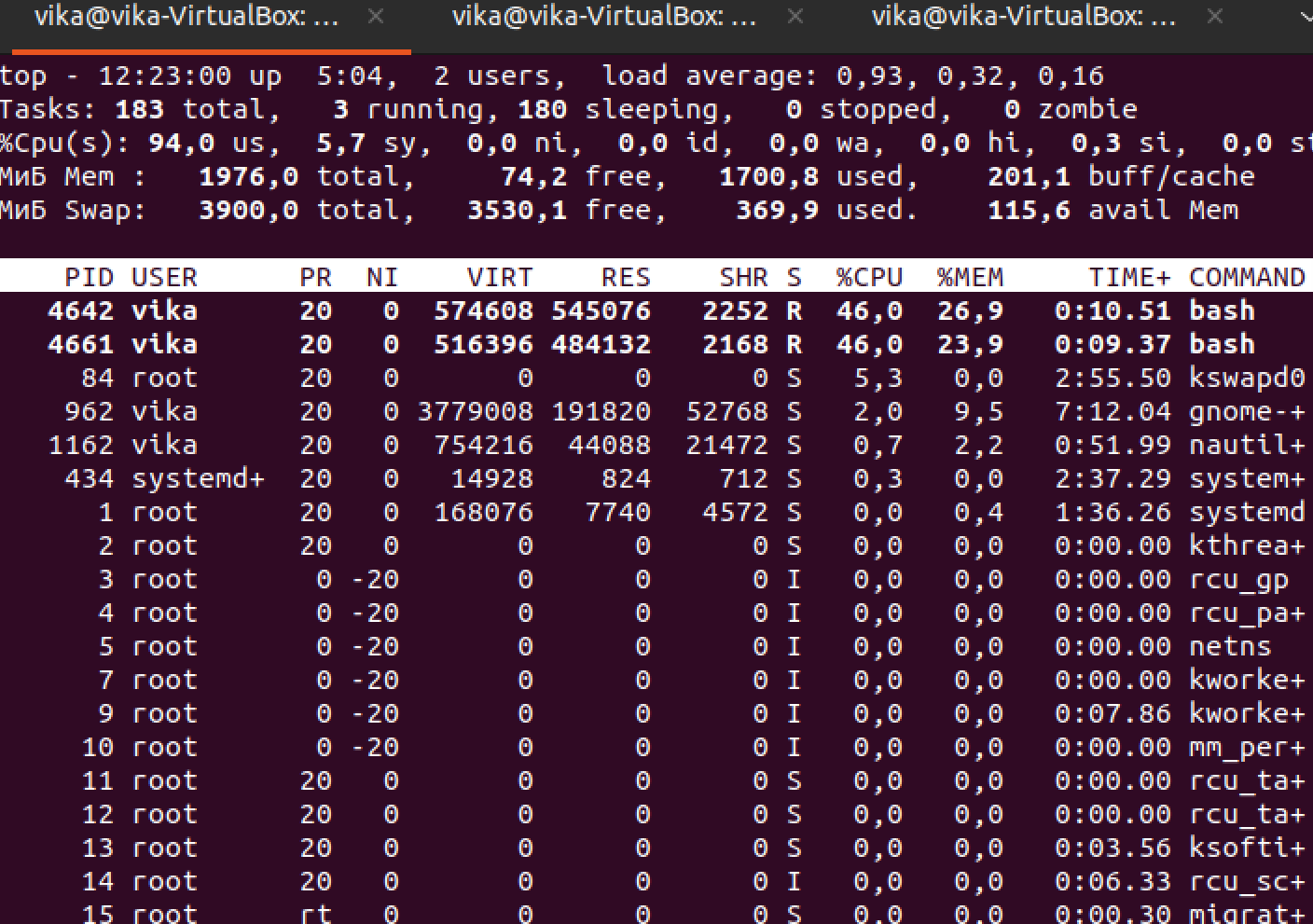


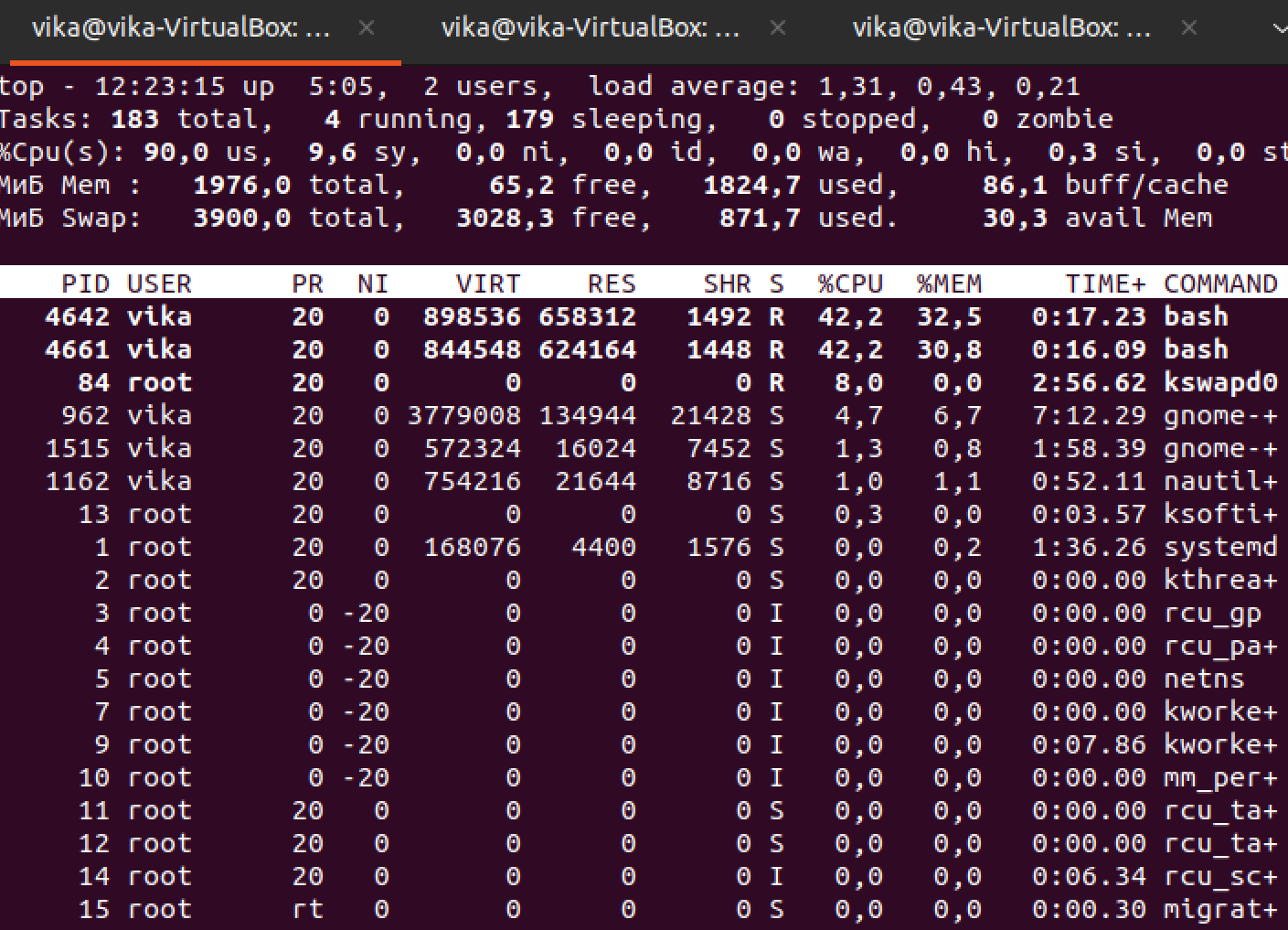
Последняя строчка в report2.log:

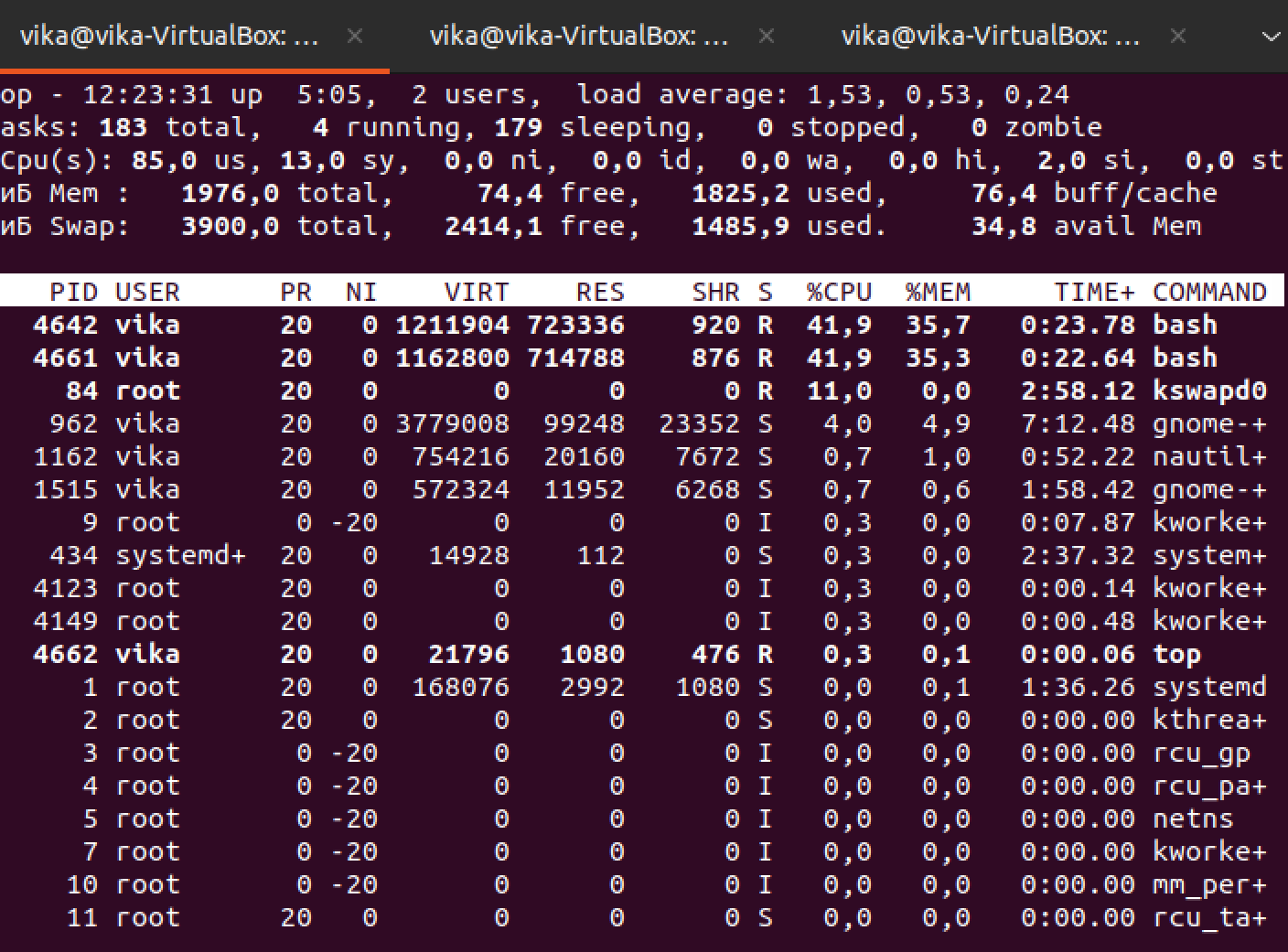


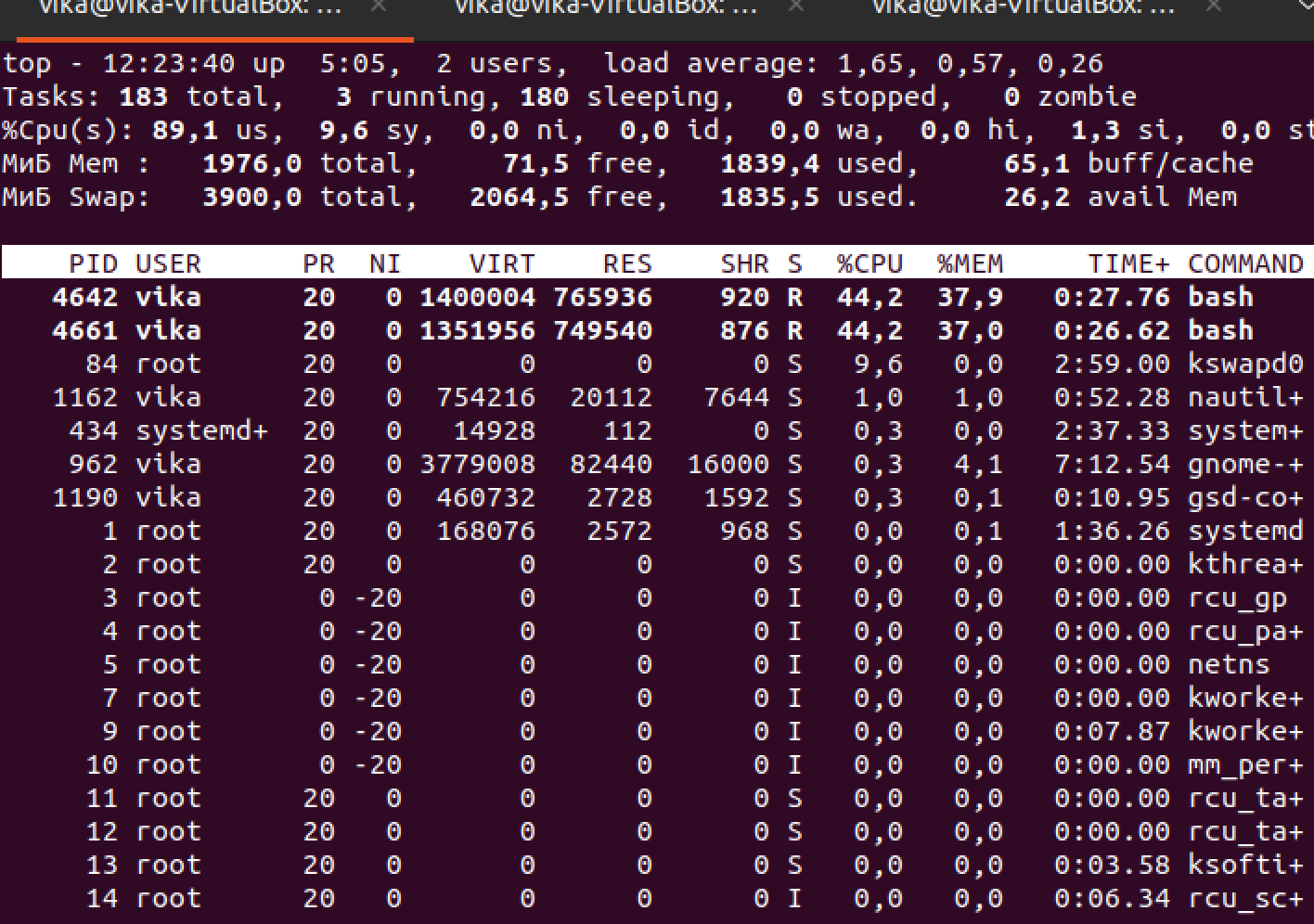


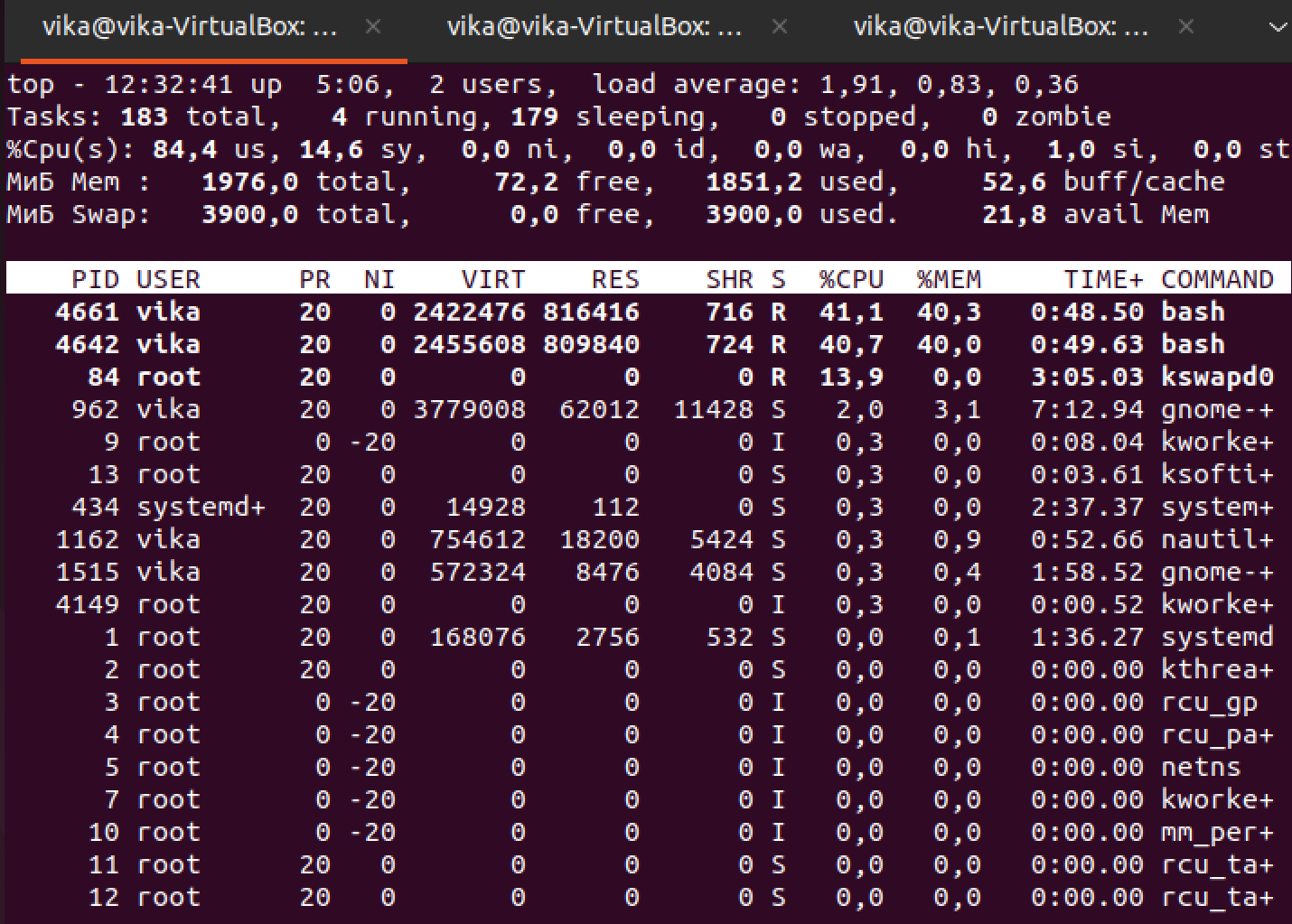


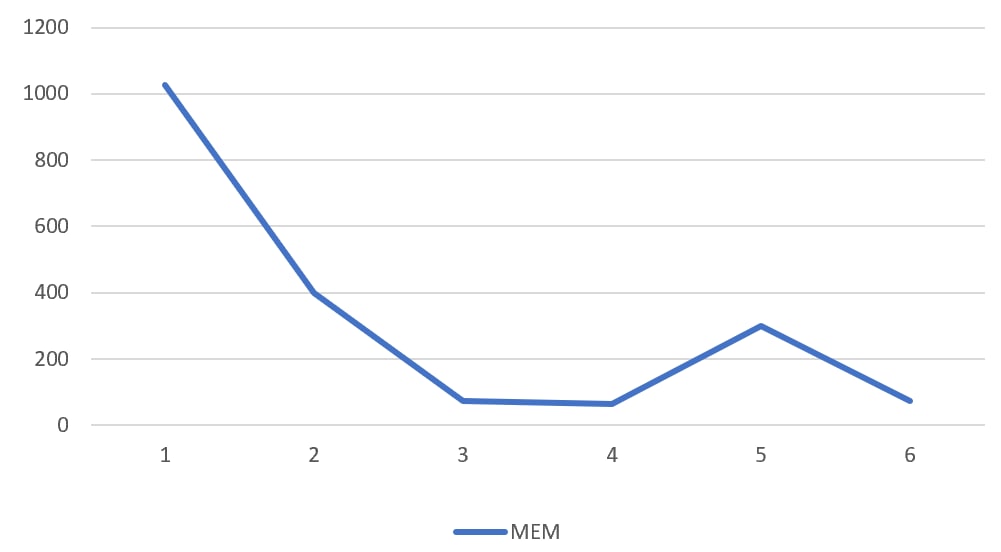












В пятерке первых процессов чаще всего встречаются процессы

gnome-+,

kswapd0,

nautil+,

сами mem.bash и mem2.bash

Вывод касаемо использования оперативной и виртуальной аналогичен выводу из первого этапа. Кардинальное отличие работы одновременно двух скриптов от работы одного заключается в том, что в момент,

когда они оба занимают по 50% оперативной и виртуальной памяти, ОС сначала завершает один из процессов. Память от завершенного процесса освобождается, после чего продолжает работать один из сĸриптов. Затем он сам заполняет всю память, что приводит к аварийному завершению.

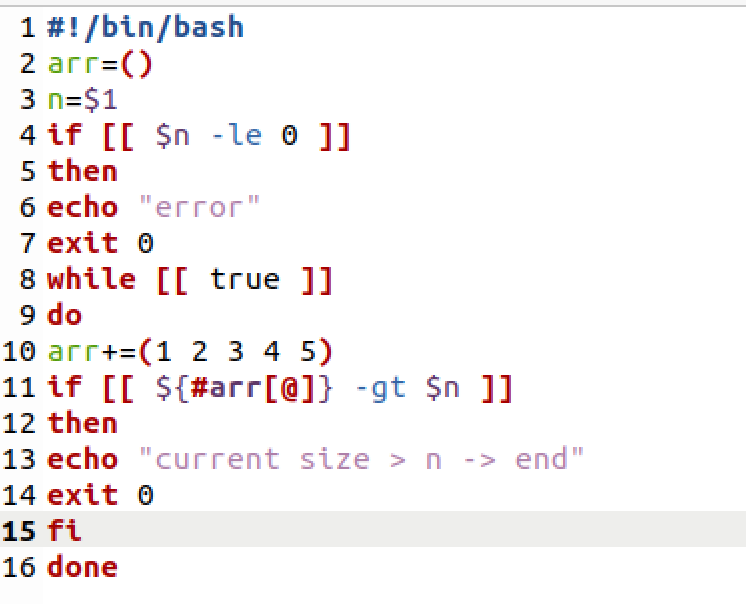
**2 эксперимент:**

Скрипт newmem.bash

Данный скрипт является измененной копией скрипта mem.bash. Он завершает работу, как только размер создаваемого массива превысит значение N, передаваемое в качестве параметра скрипту.

Задача: определить граничные значения потребления памяти, обеспечивающие безаварийную работу для регулярных процессов, запускающихся с заданной интенсивностью.

Скрипт newmem.bash:



Сначала мы bash запустили программу с параметрами: k=10 N=3100000

(N в 10 раз меньше, чем размер массива, который является максимальным для запуска одного скрипта)

Убедились в том, что все к=10 запусков успешно завершились, и в системном журнале нет записей об аварийной остановке нашего скрипта

Изменим значение k на 30 и снова запустим скрипт

В результате некоторые процессы выполнились, а некоторые завершились аварийно из-за нехватки памяти. Это объясняется тем, что в зависимости от момента, в который выполнялся какой-либо из процессов, объем оперативной памяти близился к нулю, ему могло быть недостаточно памяти или же наоборот памяти хватало (поэтому некоторые процессы не совершили аварийного завершения), так как память из подкачки поступает постепенно.

Чтобы найти N, при котором все программы с k=30 смогут выполниться, необходимо максимально возможный размер массива (31000000) разделить на 30

31000000/30=1033333,(33)

Следовательно, максимально возможное N = 1033333.