МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет информационных технологий механики и оптики

Мегафакультет информационных технологий и программирования

Факультет информационных систем и технологий

**Лабораторная работа № 5**

**По дисциплине «Операционные системы»**

**Управление памятью в ОС Linux.**

**Выполнил студент группы №M3208**

**Предыбайло Влада Евгеньевна**

**Проверил:**

**Дюкарева Вероника Максимовна**

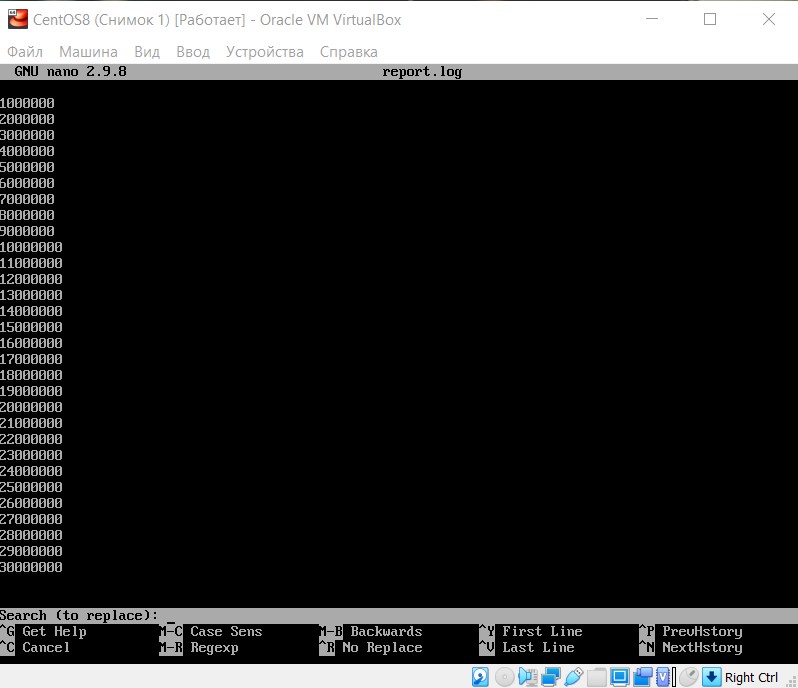
**Санкт – Петербург**

**2020**

**Эксперимент №1**

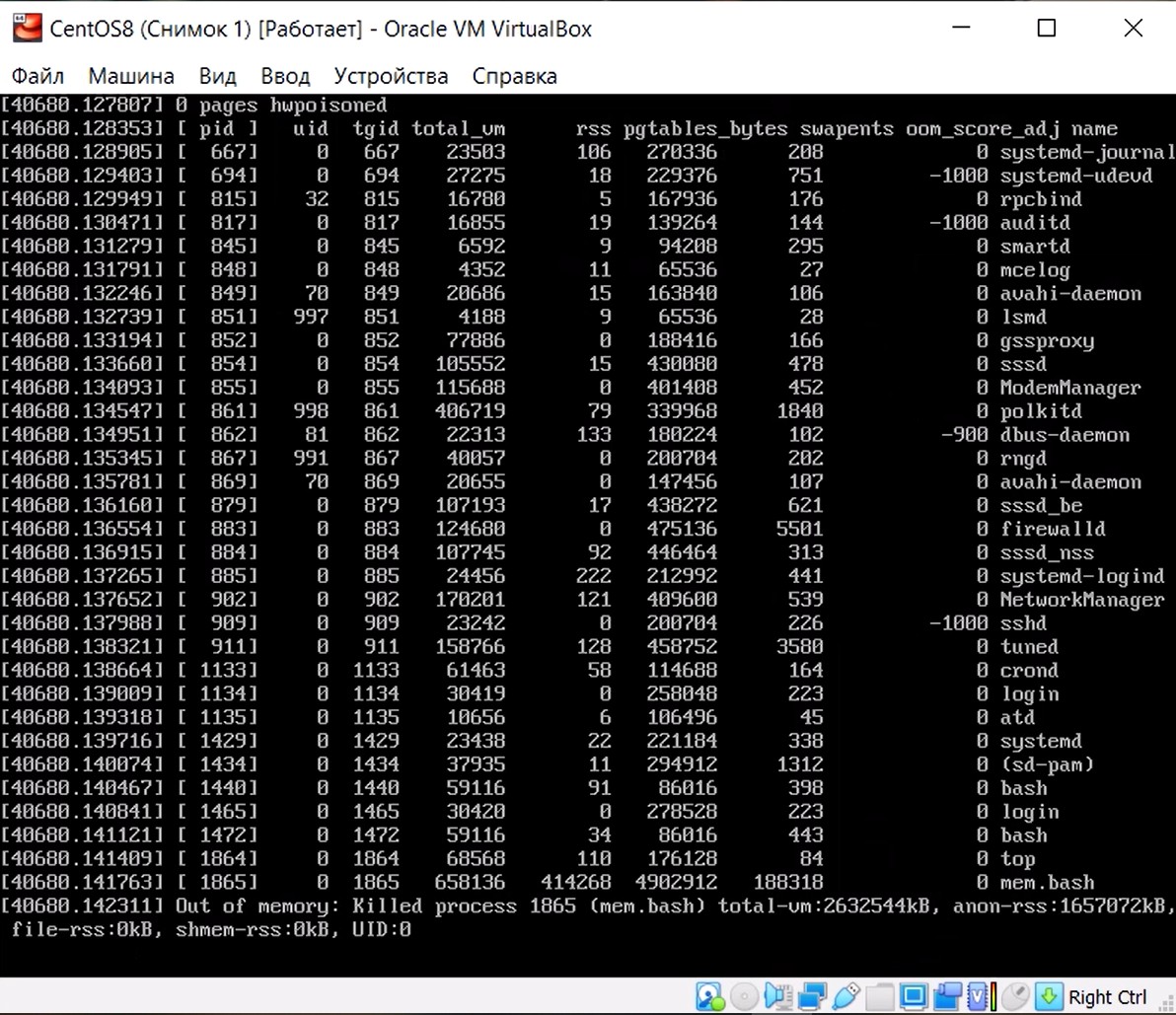
*Подготовительный этап:*

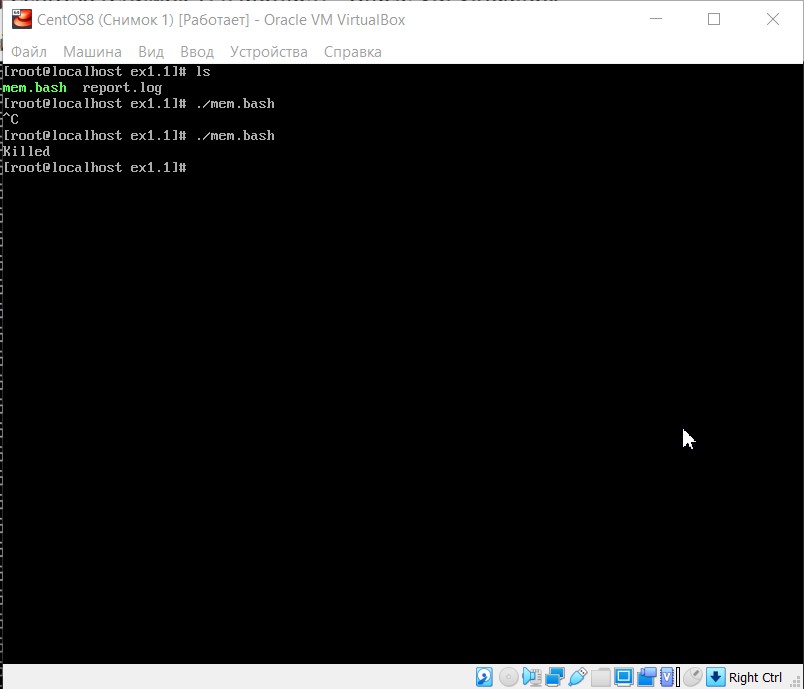
Создали скрипт mem.bash, который выполняет бесконечный цикл. На каждом шаге цикла в конец массива добавляется последовательность из 10 элементов, каждый 100000-й шаг в файл report.log записывается текущий размер массива.



*Первый этап:*

Запустить скрипт **mem.bash**, дождаться аварийной остановки и зафиксировать последние сообщения системного журнала.

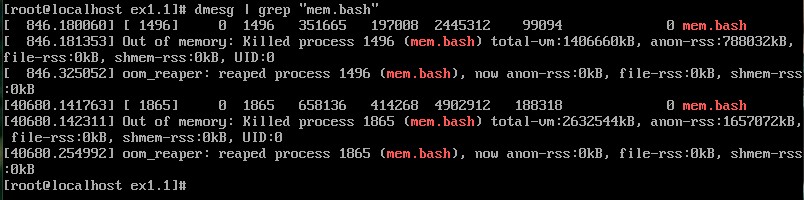




Значение последней строки в файле **report.log** – 30000000.

Отслеживаю значения с помощью команды **top** и записываю видео, оно находится в зип файле, а все значения по изменению файла внесены в таблицу

Команда **dmesg | grep “mem.bash”**



*Второй этап:*

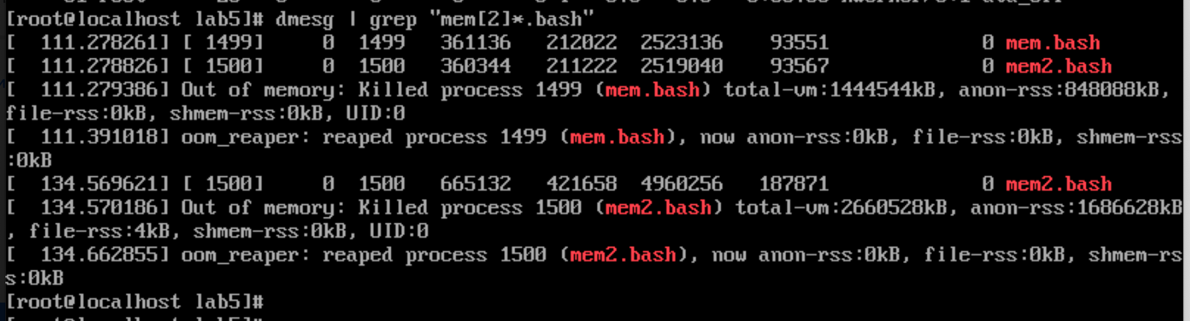
Скопировала скрипт из файла mem.bash в mem2.bash

Создала скрипт который запускает эти два файла в фоновом режиме

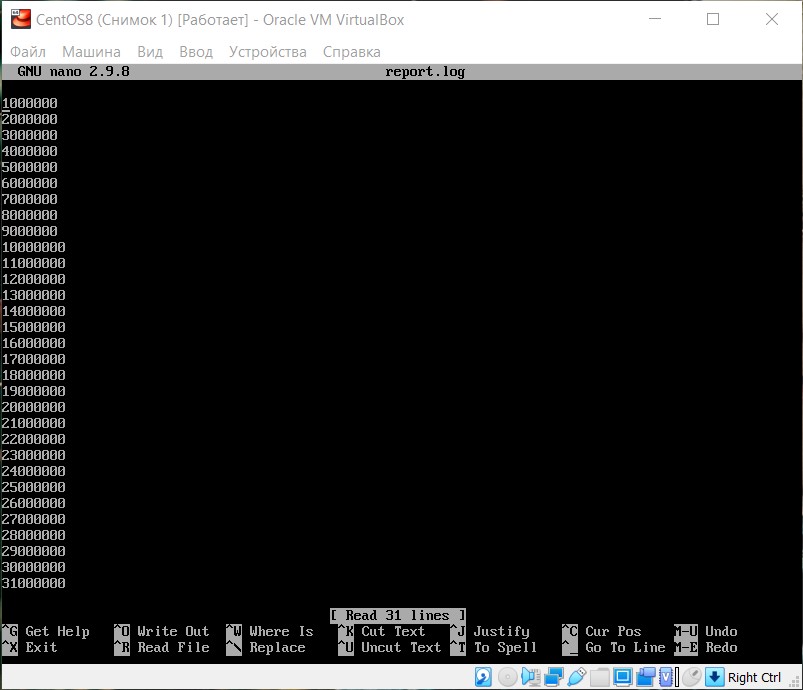


Запустила на одной консоли **top** на другой **run.sh**. Отслеживаю значения с помощью команды **top** и записываю видео, оно находится в зип файле, а все значения по изменению файла внесены в таблицу

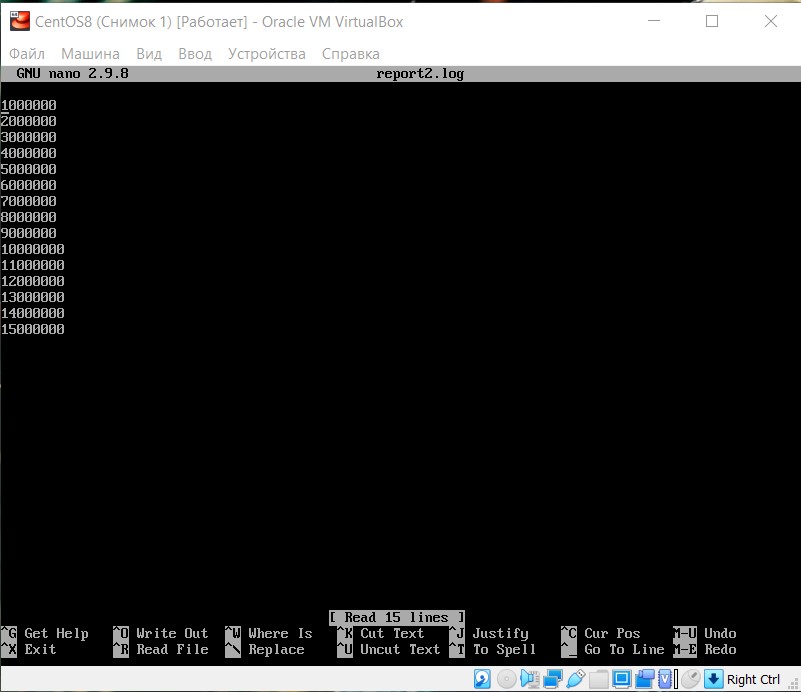
Результат команды **dmesg | grep “mem[2]\*.bash”:**

****

*Последняя строчка в файле* ***report.log:***

****

*Последняя строчка в файле* ***report2.log:***

****

*График изменения величин (первый этап):*

*Первый этап:* при запуске скрипта процесс занимает всю физическую память. На графике можно заметить, что величина свободной физической памяти постепенно уменьшается (**mem free**), а размер свободной памяти файла подкачки постоянен (**swap free**). Когда установленный объем физической памяти заканчивается, начинает использоваться выделенная область из файла подкачки. То есть уменьшается величина **swap free**.

Когда свободного места не остается и в файле подкачки, происходит аварийная остановка. (размер массива в этот момент указан выше)

*График изменения величин (второй этап):*

*Второй этап:* когдапри одновременном запуске двух скриптов, заканчивается память в **swap free**, происходит аварийная остановка первого скрипта (**mem.bash**, величину массива в этот момент можно увидеть выше). После чего вся свободная память выделена под **mem2.bash**. При нехватке памяти так же, как и на первом этапе происходит аварийная остановка теперь уже **mem2.bash** (величину массива в этот момент можно увидеть выше).

**Эксперимент №2**

*Подготовительный этап:*

Создали копию скрипта **mem.bash**. Изменили копию, теперь она завершает работу, как только размер создаваемого массива превысит значение **N**, передаваемого в качестве параметра скрипту. Так же убрали запись данных в файл.

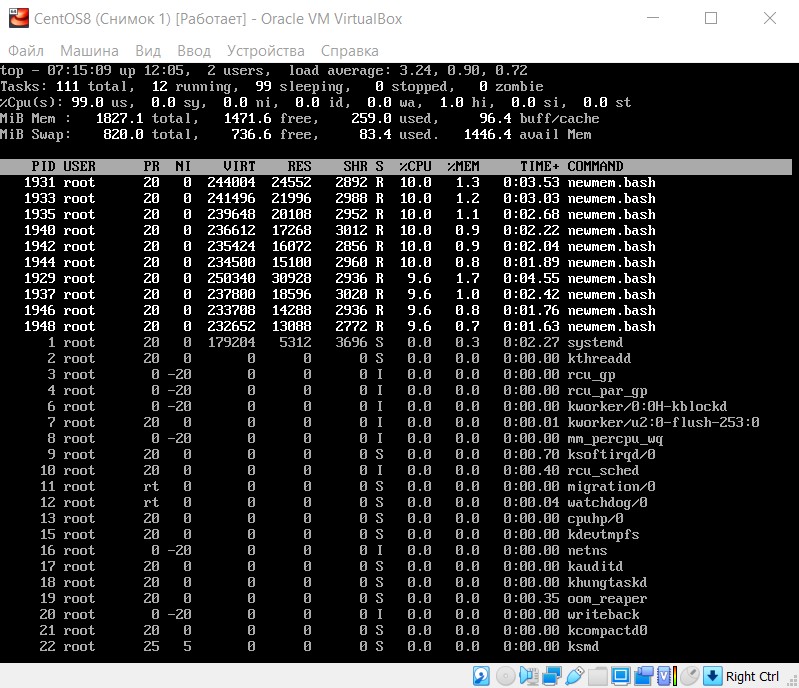
*Основной этап:*

Создали скрипт, который запускает **newmem.bash** каждую секунду, используя один и тот же параметр **N** так, что всего будет осуществлено **K** запусков.

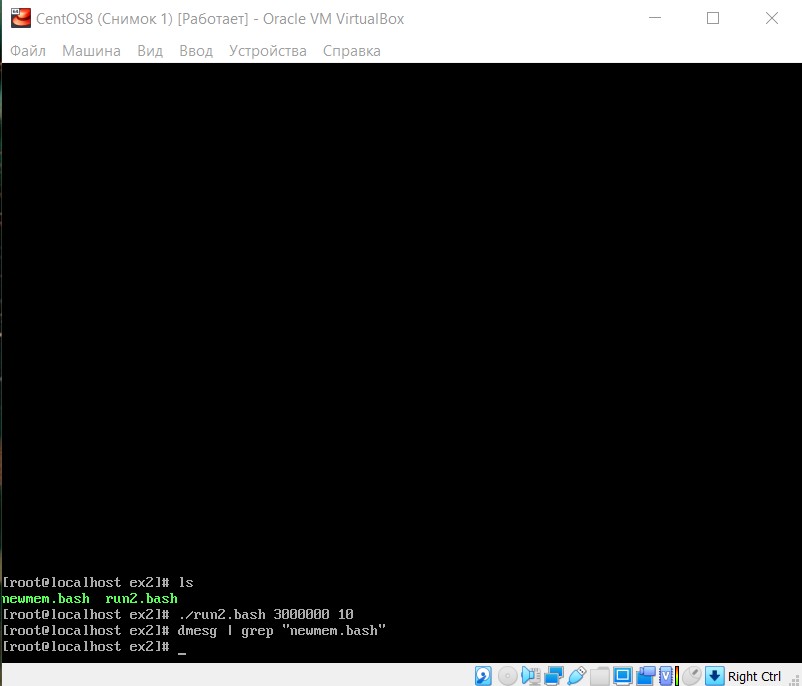
Установили значение **N** = 30000000/10 = **3000000**

Значение **K** = **10**

Убедились, что было запущено ровно 10 скриптов:



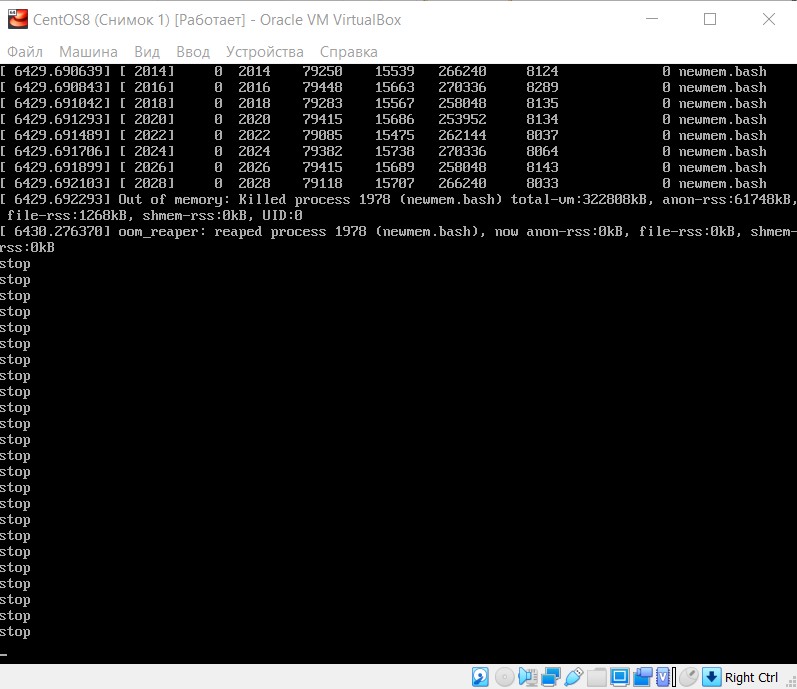
Убедились, что все 10 запущенных скриптов успешно завершились и в системном журнале нет записей об аварийных остановках:



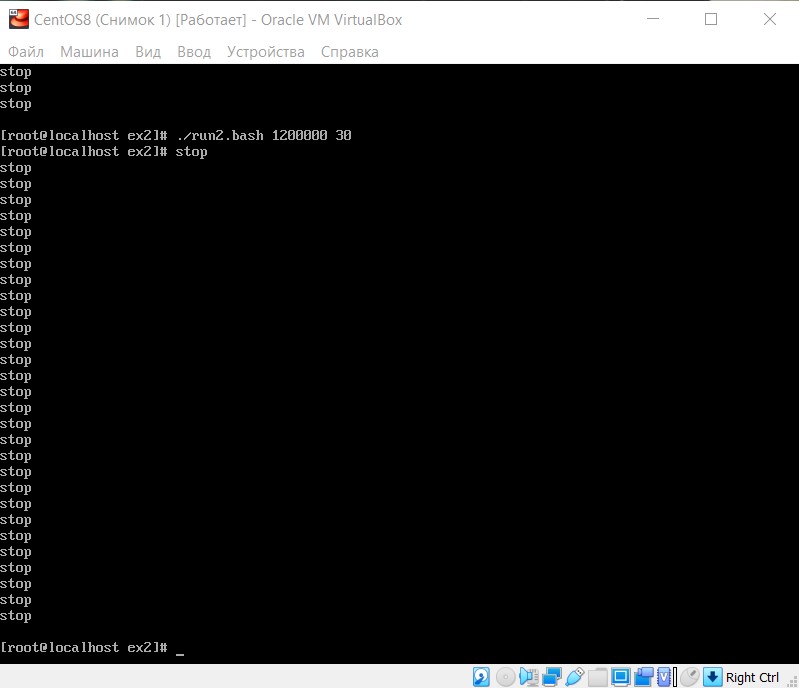
В том случае, если N = 3000000, а K = 30, происходит аварийное завершение части процессов. Объясняется это нехваткой памяти.

В первом эксперименте мы выяснили, что максимальная сумма хранимых элементов массива равна 30000000.

Когда мы создаем 30 процессов с размерами массивов 3000000, суммарное число элементов всех массивов в какой-то момент превысит 30000000, что и приведет к аварийному завершению некоторых процессов.



Максимальное значение N = 1200000, в этом случае при K = 30 не будет происходить аварийных завершений процессов



Вывод

На практике убедилась, что во время работы с оперативной памятью данные заполняют физическую память, пока не дойдут до критического значения. Далее ОС начинает заполнять файлы подкачки. В случае, когда файлов подкачки будет не хватать – произойдет аварийная остановка процесса.