**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**ОТЧЕТ**

по лабораторной работе №5 «Управление памятью в ОС Linux»

по дисциплине «**Безопасность жизнедеятельности**»

Автор:

*Урбанович Рихард*

Факультет:

*ФИТиП*

Группа:

*М3202*



Санкт-Петербург 2020

**Данные о конфигурации системы:**

Общий объём оперативной памяти (Mem): 1827,1 Мб

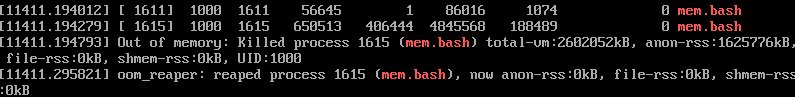
Объём раздела подкачки (Swap): 820 Мб

Размер страницы виртуальной памяти: 4 Кб

**Эксперимент 1.**

*Первый этап:*

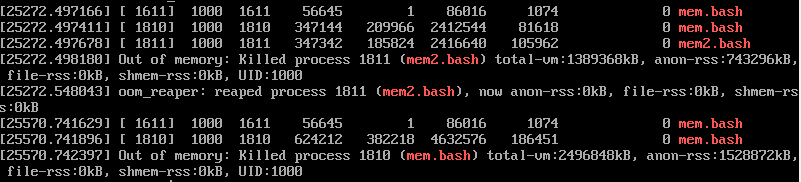
1. Записи об аварийной остановке скрипта



1. Значение в последней строке в файле **report.log**: 30’400’000.
2. Графики зависимостей свободной оперативной памяти и раздела подкачки от времени:
3. Зафиксированные значения из перечня процессов в **top** находятся в lab5.xlsx

*Второй этап:*

1. Записи об аварийной остановке скриптов

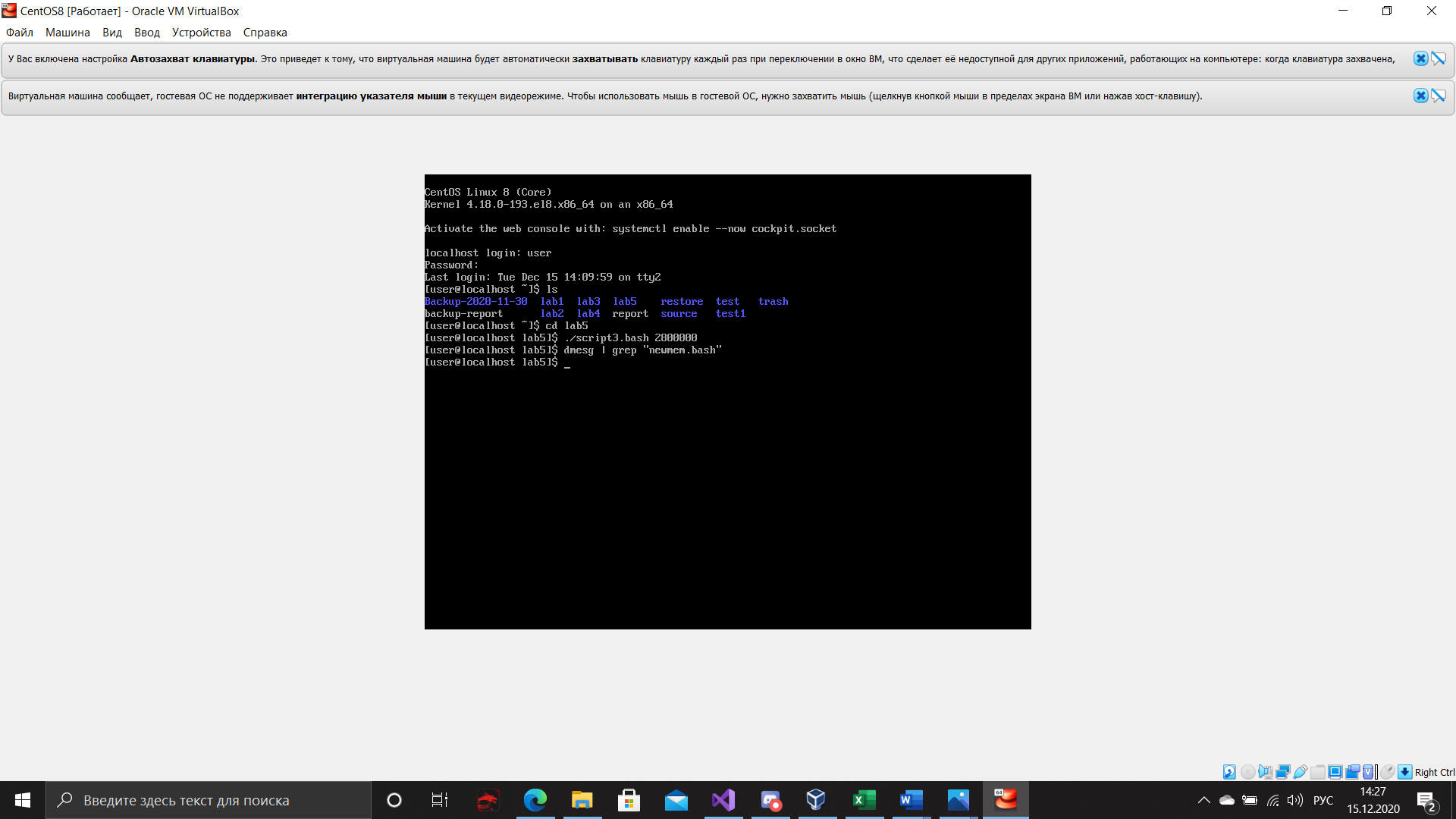


1. Значение в последней строке в файле **report.log**: 29’100’000  
   Значение в последней строке в файле **report2.log**: 14’900’000
2. Графики зависимостей процентов занимаемой памяти первым и вторым скриптом от времени:
3. Графики зависимостей занятой оперативной памяти и раздела подкачки
4. Зафиксированные значения из перечня процессов в **top** находятся в lab5.xlsx

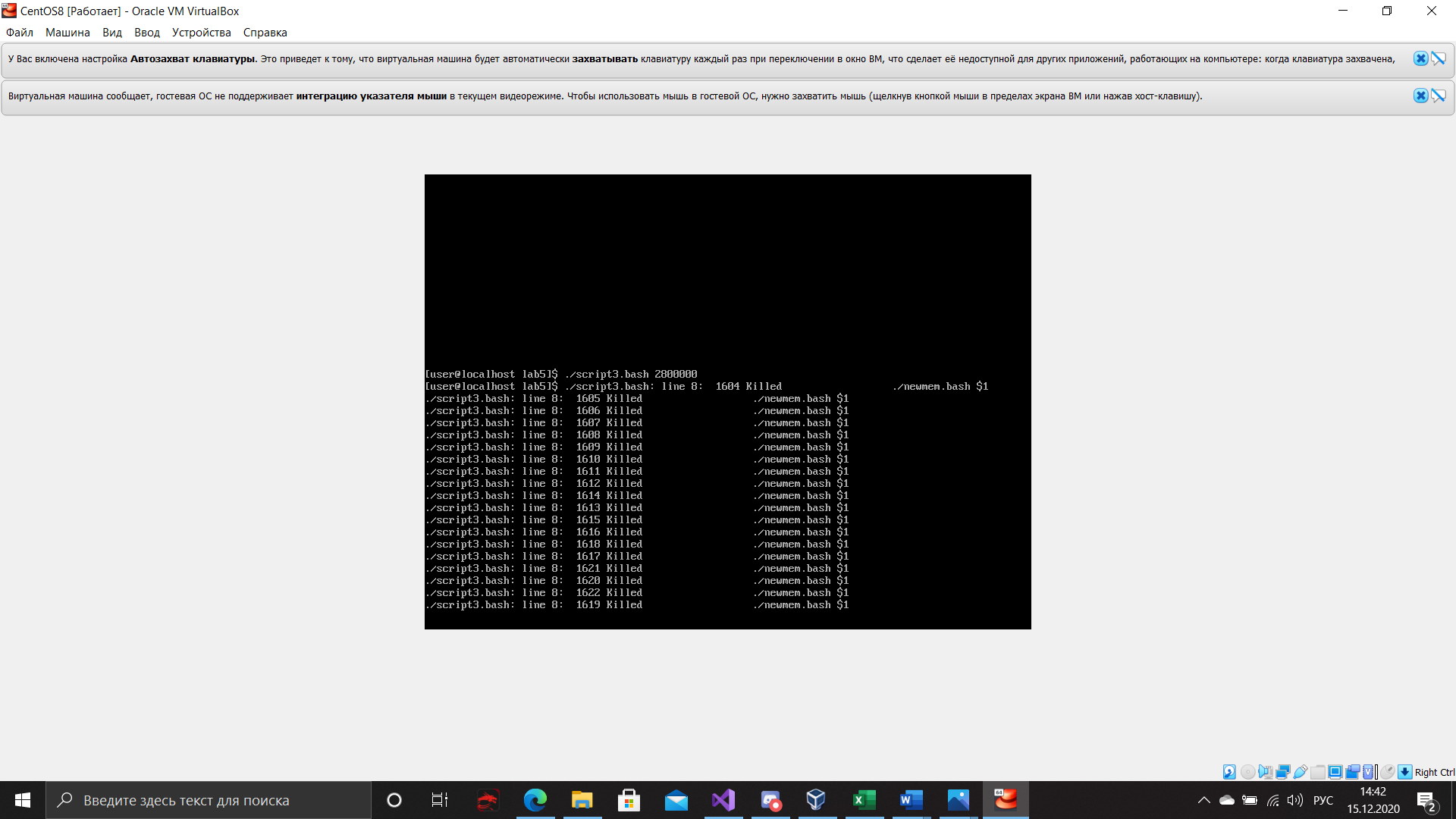
**Выводы:** по графикам и зафиксированным значениям мы можем отметить, что сначала система полностью загружает память. Когда оперативная память заканчивается, то ОС начинает использовать раздел подкачки для хранения данных. Когда раздел подкачки заканчивается и память становится полностью загруженной, то для предотвращения аварийной остановки всей системы ОС осуществляет аварийную остановку процессов.  
Если запущено несколько процессов, сильно использующих память, то останавливается только один процесс, а другим даётся возможность использовать освободившуюся память для попытки успешно закончить процесс.

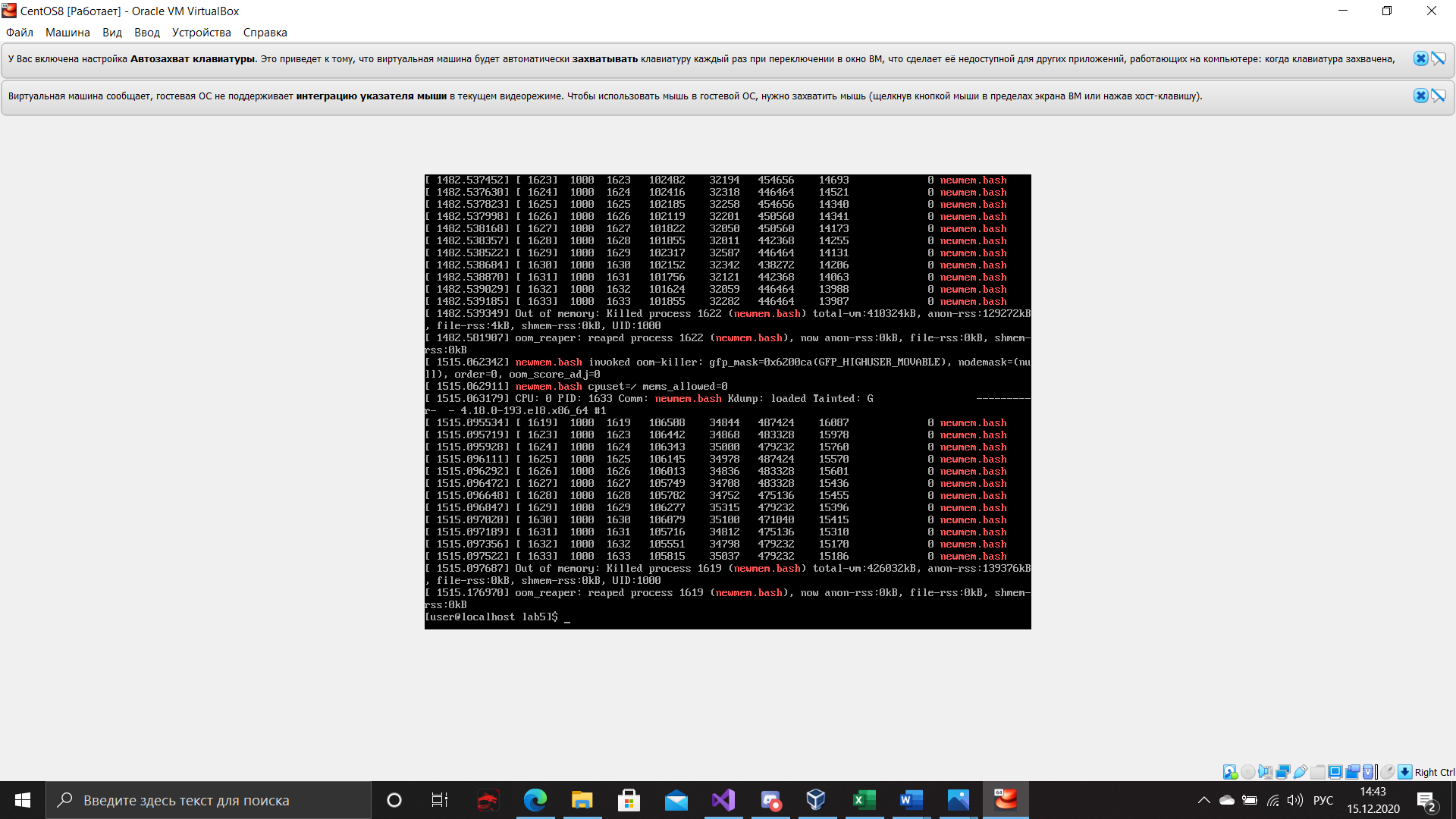
**Эксперимент 2.**

1. Установили значения K = 10 и N = 2’700’000. Убедились, что запустились все 10 процессов были запущены и все процессы завершили свою работу успешно.

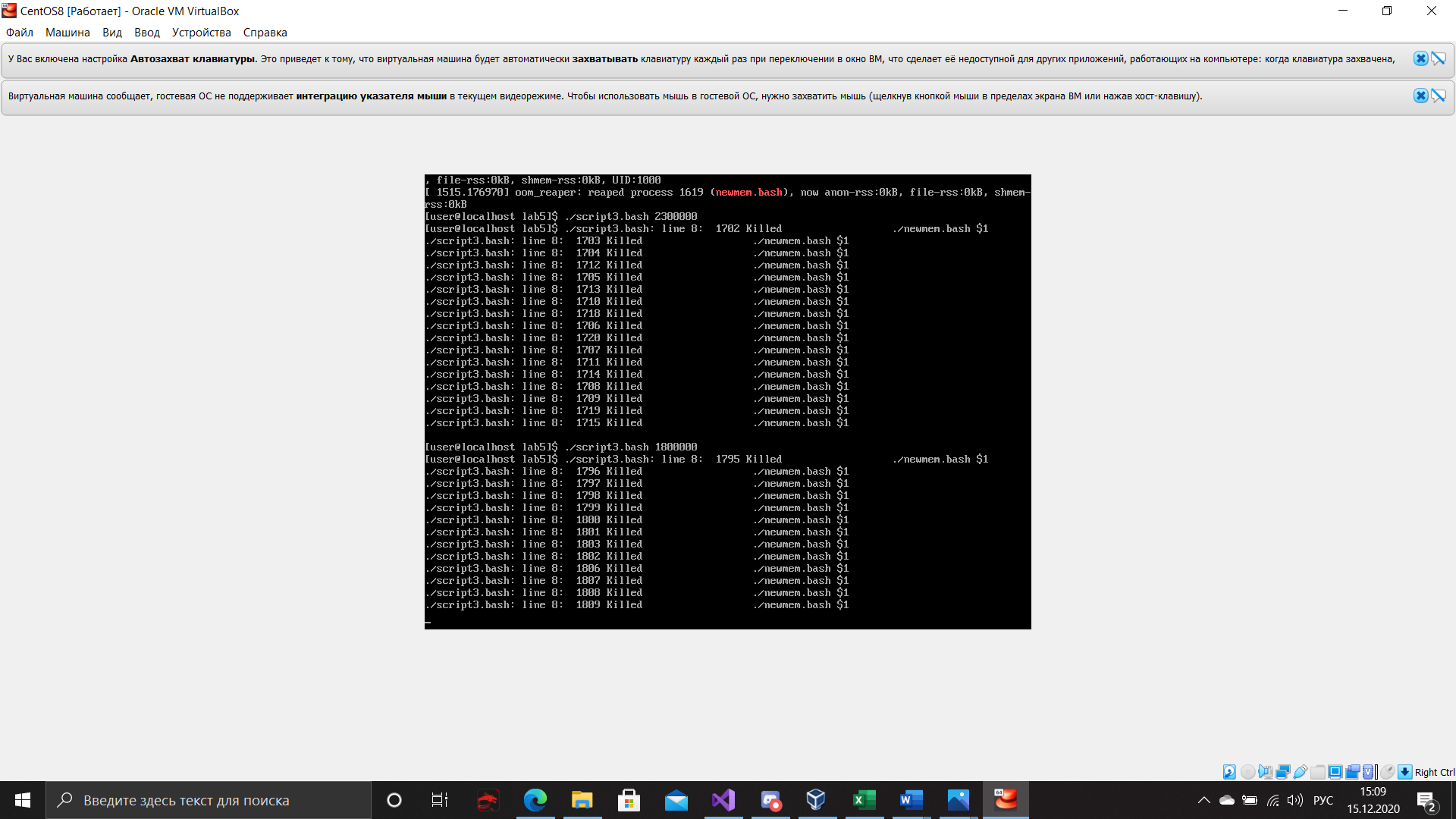


1. Установили значения K = 30 и N = 2’700’000. Убедились, что все 30 процессов были запущены и 18 процессов завершили свою работу аварийно.

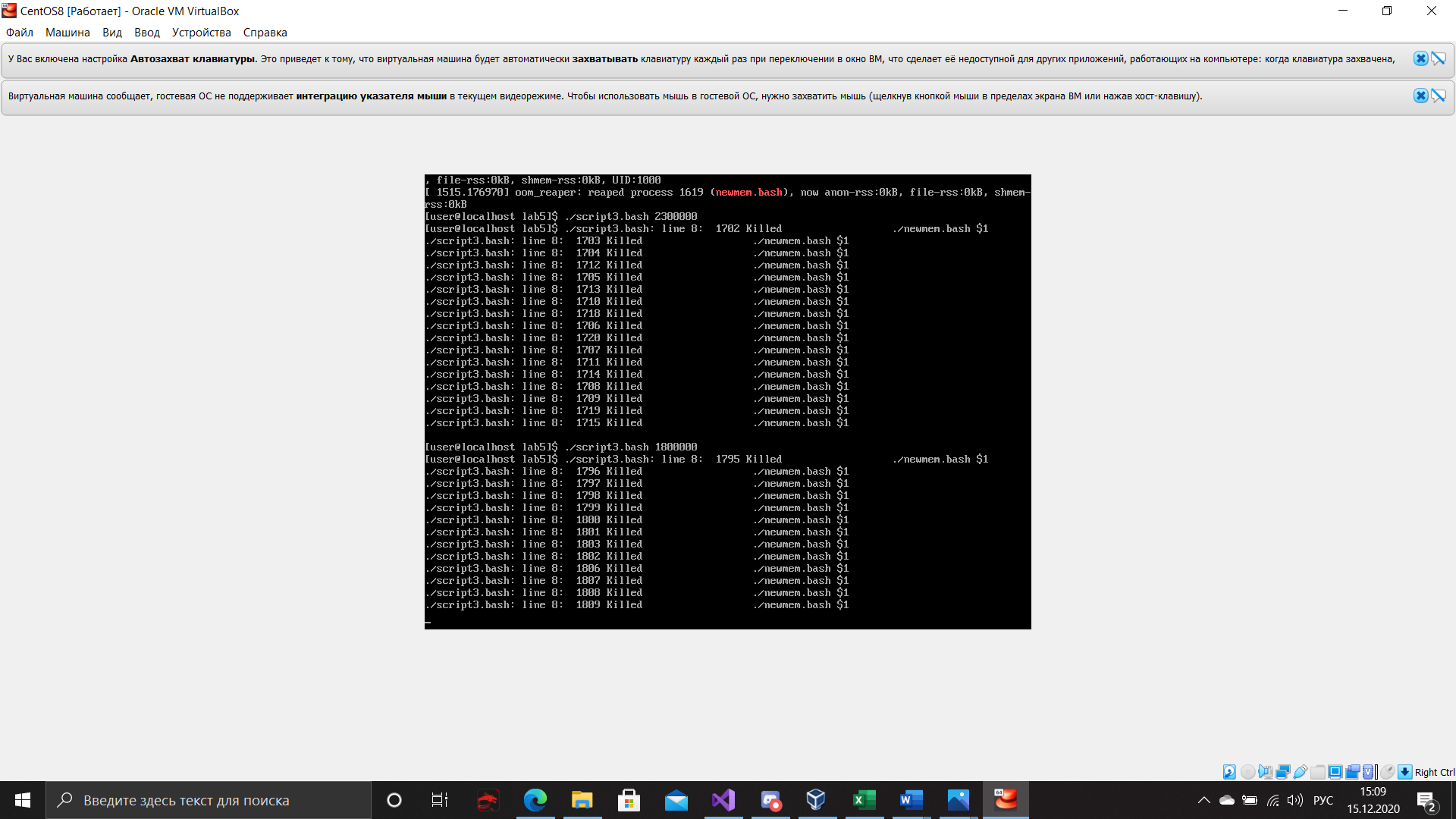




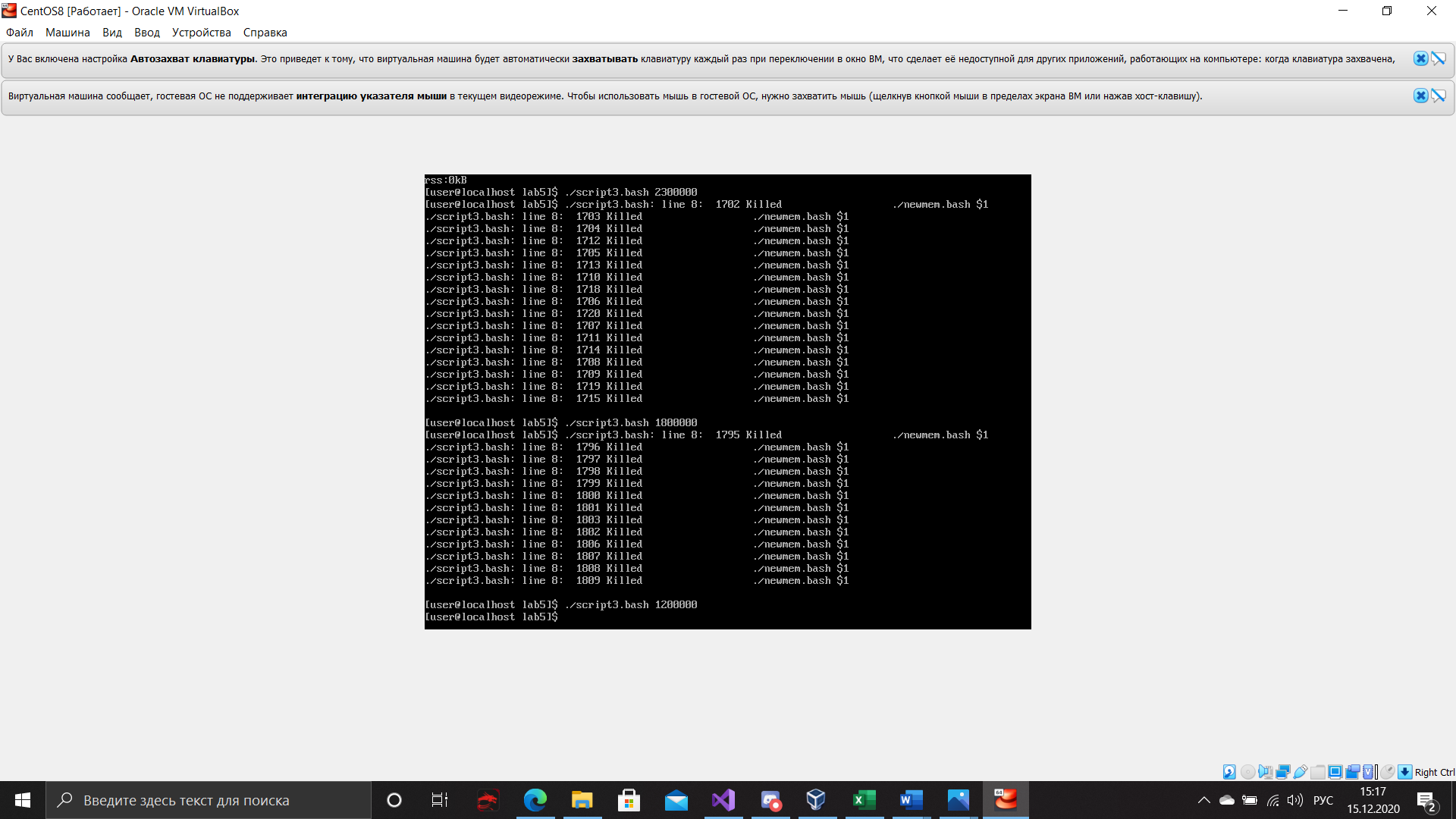
1. Установили значения K = 30 и N = 2’300’000. Убедились, что все 30 процессов были запущены и 16 процессов завершили свою работу аварийно.



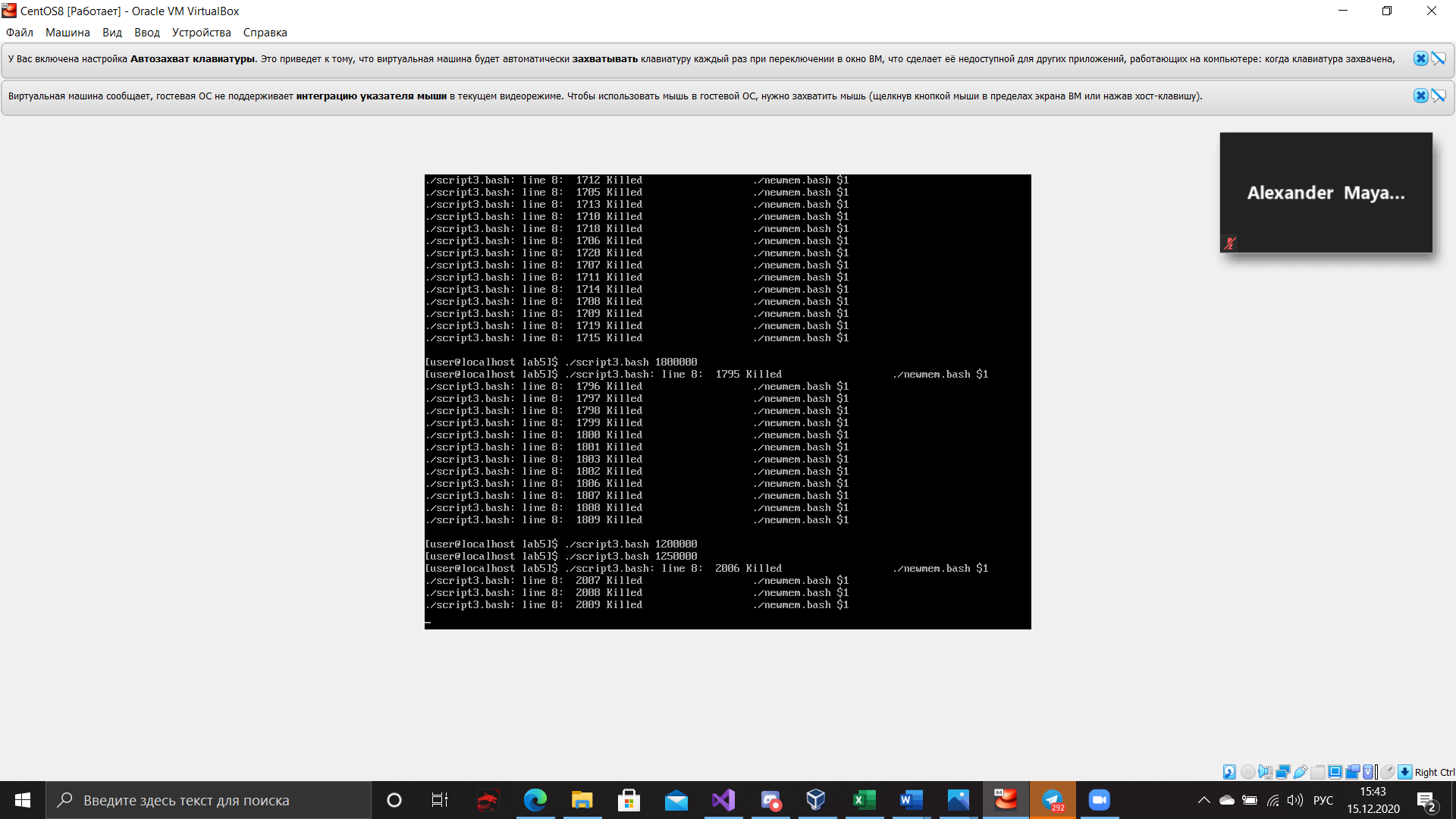
1. Установили значения K = 30 и N = 1’800’000. Убедились, что все 30 процессов были запущены и 12 процессов завершили свою работу аварийно.



1. Установили значения K = 30 и N = 1’200’000. Убедились, что все 30 процессов были запущены и все процессы завершили свою работу успешно.



1. Установили значения K = 30 и N = 1’250’000. Убедились, что все 30 процессов были запущены и 3 процесса завершили свою работу аварийно.



**Вывод:** при К = 30: N = 1’200’000. Чем больше процессов одновременно работают, тем при меньшем N они смогут завершить свою работу успешно. Также можно отметить зависимость, что если K = const, то чем больше N – тем больше процессов завершают работу аварийно. Связано это с тем, что каждому процессу требуется больше времени и памяти, а освобождается память только в случае, если какой-нибудь процесс уже завершил свою работу.