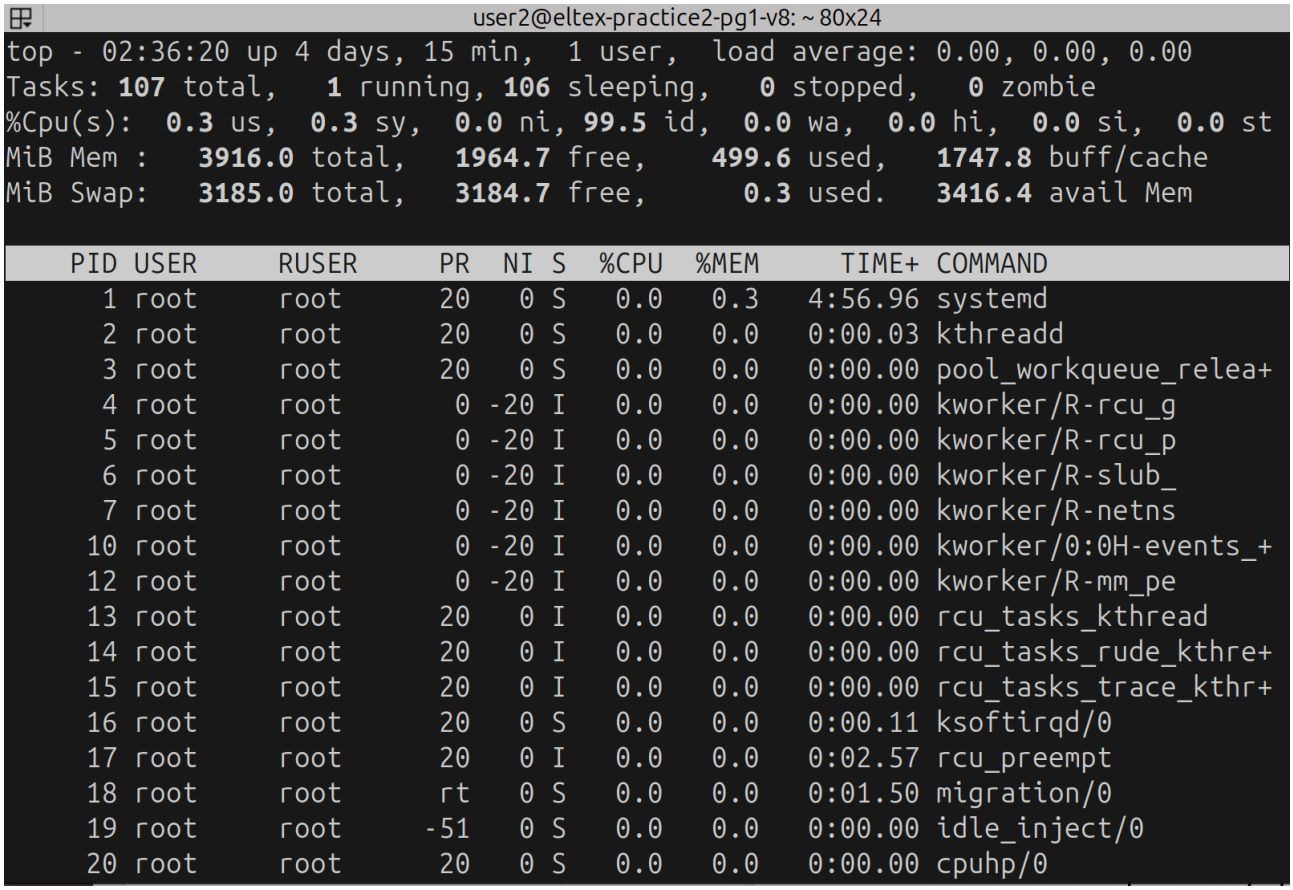
1.Войдите под пользователем user1 из практики 2 (su - user1)

2.Подсчитайте количество процессов, имеющих несколько потоков выполнения
ps -eLo pid | sort | uniq -c | awk '$1 > 1 {print}' | wc -l
10

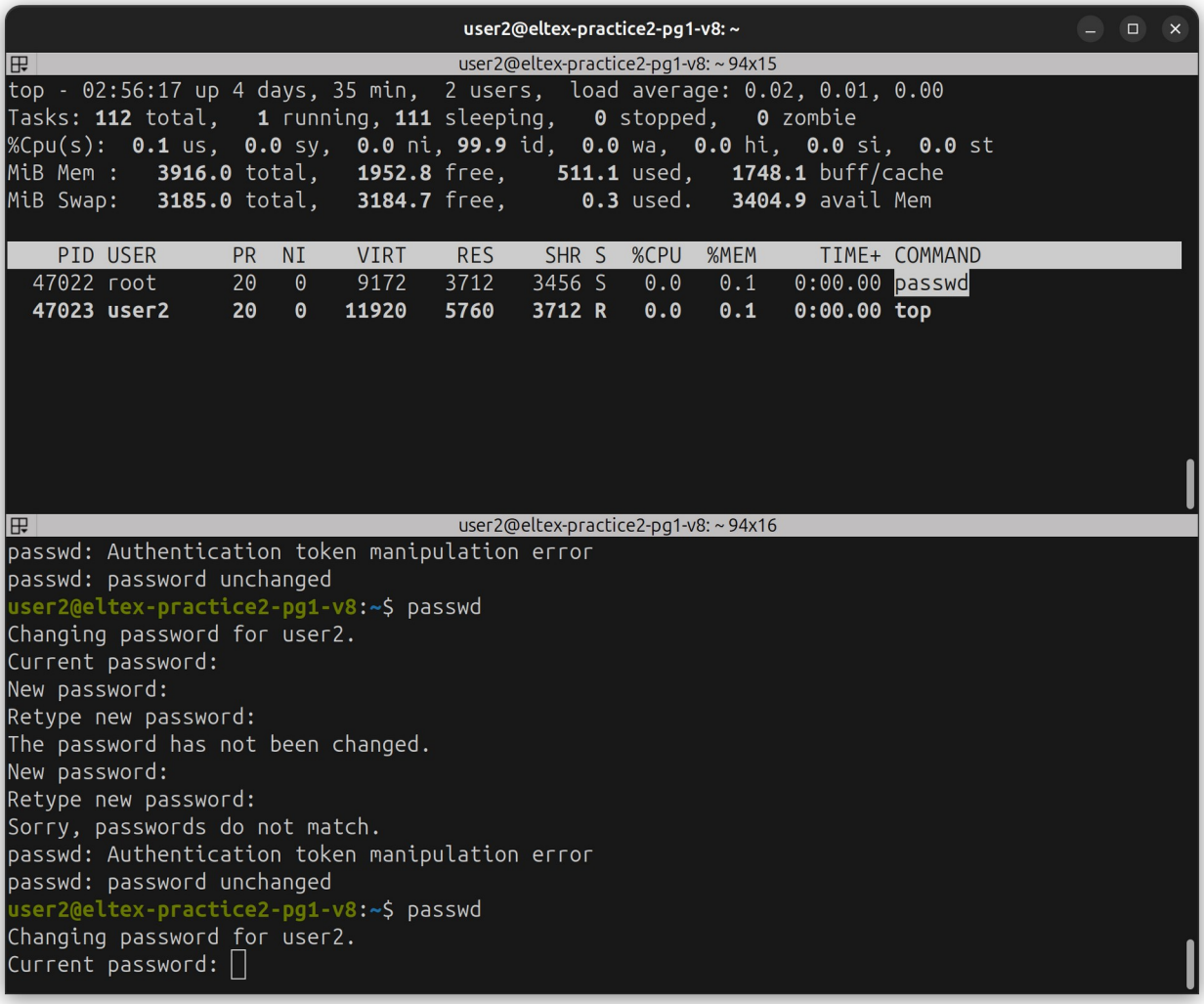
3. Запустите top и настройте вывод полей с информацией о процессе следующим образом:



4. В другом терминальном окне выполните команду passwd и оставьте ее в состоянии запроса текущего пароля

5. Перейдите в терминальное окно с top и выполните следующие действия:
• выведите все процессы, для которых реальным пользователем является пользователь, которым вы вошли в сеанс;
• найдите процесс, запущенный командой passwd;
• отправьте этому процессу сигналы 15 (SIGTERM), 2 (SIGINT), 3 (SIGQUIT), 9(SIGKILL)

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
46812 user2     20   0    8776   5504   3840 S   0.0   0.1   0:00.02 bash
46983 user2     20   0    8644   5504   3840 S   0.0   0.1   0:00.01 bash
47024 user2     20   0   11920   5760   3712 R   0.0   0.1   0:00.00 top
```



kill -15 PID
kill -2 PID
kill -3 PID
kill -9 PID

passwd убил только SIGKILL

6. Выполните команду vim ~/file_task3.txt и нажмите Ctrl-Z

```
user2@eltex-practice2-pg1-v8:~$ vim ~/file_task3.txt

[1]+  Stopped                 vim ~/file_task3.txt
user2@eltex-practice2-pg1-v8:~$
```

7. Выполните команду sleep 600, нажмите Ctrl-Z и выполните команду jobs

```
user2@eltex-practice2-pg1-v8:~$ sleep 600
^Z
[2]+  Stopped                 sleep 600
user2@eltex-practice2-pg1-v8:~$ jobs
[1]-  Stopped                 vim ~/file_task3.txt
[2]+  Stopped                 sleep 600
user2@eltex-practice2-pg1-v8:~$
```

8. Последнее задание (sleep 600) сделайте фоновым

bg %2

9. Измените число NICE у задания (sleep 600), сделав его равным 10

10 Проверьте, что число NICE у этого задания изменилось

```
user2@eltex-practice2-pg1-v8:~$ pgrep -a sleep
47070 sleep 600
user2@eltex-practice2-pg1-v8:~$ renice -n 10 47070
47070 (process ID) old priority 0, new priority 10
user2@eltex-practice2-pg1-v8:~$
```

```
[2] 47093
user2@eltex-practice2-pg1-v8:~$ renice -n 10 47093
47093 (process ID) old priority 0, new priority 10
user2@eltex-practice2-pg1-v8:~$ ps axl | grep sleep
0  1002   47093   47048  30  10   5684  2048 hrtime SN   pts/0      0:00 sleep
0  1002   47096   47048  20   0   6544  2304 pipe_r S+   pts/0      0:00 grep
ep
user2@eltex-practice2-pg1-v8:~$
```

11 Сделайте задание vim ~/file_ task3.txt активным и выйдите из редактора

fg %1

12 Отправьте сигнал 15 (SIGTERM) заданию sleep 600 и выполните команду jobs

```
user2@eltex-practice2-pg1-v8:~$ kill -15 47105
user2@eltex-practice2-pg1-v8:~$ jobs
[3]+  Stopped                 vim ~/file_task3.txt
[4]-  Terminated              sleep 600
user2@eltex-practice2-pg1-v8:~$ jobs
[3]+  Stopped                 vim ~/file_task3.txt
user2@eltex-practice2-pg1-v8:~$
```

13 Создайте перехватчик сигналов SIGINT и SIGQUIT внутри командного интерпретатора, который выводит сообщение «Меня голыми руками не возьмёшь!» (используйте встроенную команду trap) и отправьте сигналы самому себе

```
user2@eltex-practice2-pg1-v8:~$ trap 'echo "Меня голыми руками не возьмёшь!"' SIGINT SIGQUIT
Меня голыми руками не возьмёшь! ^C

Меня голыми руками не возьмёшь! ^\

user2@eltex-practice2-pg1-v8:~$
```

Раздел 2

1. Создайте скрипт на языке bash с именем template_task.sh,

```
#!/bin/bash

script_name=$(basename "$0")

if [ "$script_name" = "template_task.sh" ]; then
    echo "я бригадир, сам не работаю"
    exit 1
fi

log_file="report_${script_name%.*}.log"  # Удаляем расширение .sh если есть

echo " $(date '+%Y-%m-%d %H:%M:%S') Скрипт запущен" >> "$log_file"

random_seconds=$(( RANDOM % 1771 + 30 ))  # 1800-30+1=1771

sleep $random_seconds

minutes=$(( random_seconds / 60 ))

echo " $(date '+%Y-%m-%d %H:%M:%S') Скрипт завершился, работал $minutes минут" >> "$log_file"

exit 0
```

2. Создайте скрипт на языке bash с именем observer.sh

Observer.sh

```
#!/bin/bash

CONFIG_FILE="observer.conf"
LOG_FILE="observer.log"

if [ ! -f "$CONFIG_FILE" ]; then
    echo "$(date '+%Y-%m-%d %H:%M:%S') [ERROR] Конфигурационный файл $CONFIG_FILE не найден" >
    exit 1
fi

is_running() {
    script_name=$1
    for pid in /proc/[0-9]*; do
        if [ -f "$pid/cmdline" ]; then
            if grep -q "$script_name" "$pid/cmdline" 2>/dev/null; then
                return 0
            fi
```

```
        fi
    done
    return 1
}

while IFS= read -r script || [ -n "$script" ]; do
    script_name=$(basename "$script")
    if is_running "$script_name"; then
        echo "$(date '+%Y-%m-%d %H:%M:%S') [INFO] Процесс $script_name уже запущен" >> "$LOG_>
    else
        nohup "$script" >/dev/null 2>&1 &
        echo "$(date '+%Y-%m-%d %H:%M:%S') [ACTION] Запущен процесс $script_name (PID: $!)" >>
    fi
done < "$CONFIG_FILE"

exit 0
```

3. Настройте запуск observer.sh посредством cron по расписанию – 1 раз в минуту

crontab -e

* * * * * /home/user2/observer.sh

4.Создайте несколько символьных ссылок на файл template_task.sh сразличными именами (рабочие задачи), добавьте в файл конфигурации observer.conf соответствующие записи об этих задачах, включая исходный файл template_task.sh

```
ln -s template_task.sh script1.sh

...
```

5. Соберите статистику работы в виде набора файлов report_*.log, observer.log, приложите их вместе с исходными текстами скриптов в качестве отчета в виде сжатого архива tar. Не забудьте остановить процесс, удалив задачу в cron!

```
mkdir observer_report

cp /path/to/scripts/observer.* .

…

tar -czf observer_report.tar.gz *
```