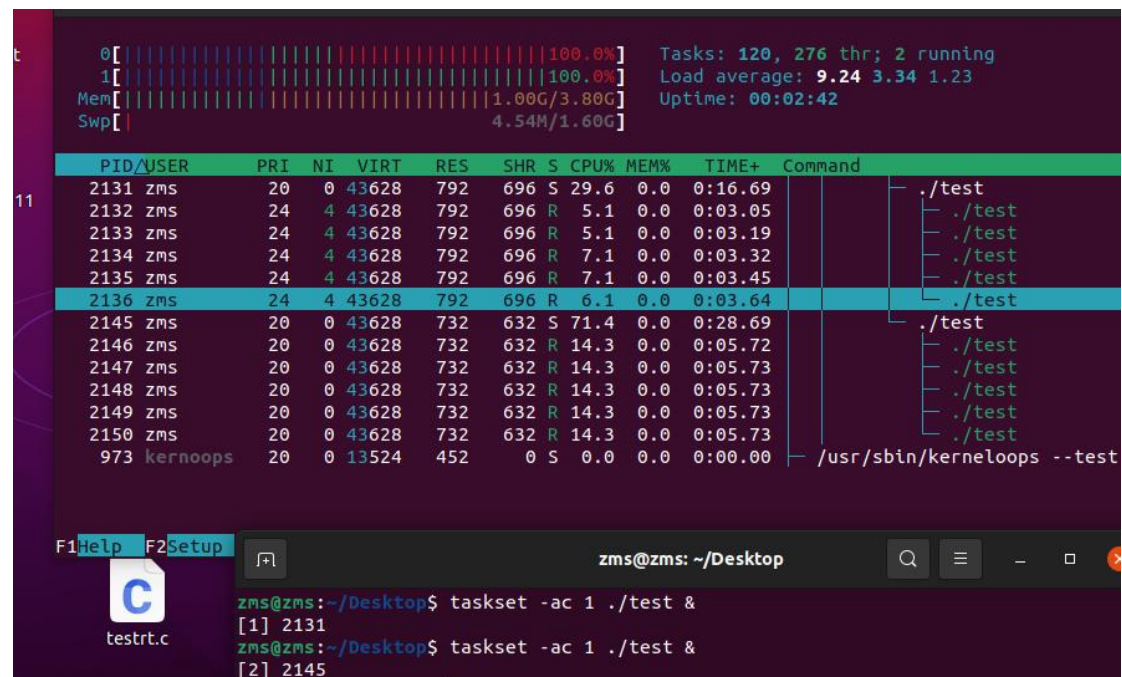


Task 1.1

将五个线程 ni 设置为 4 即可

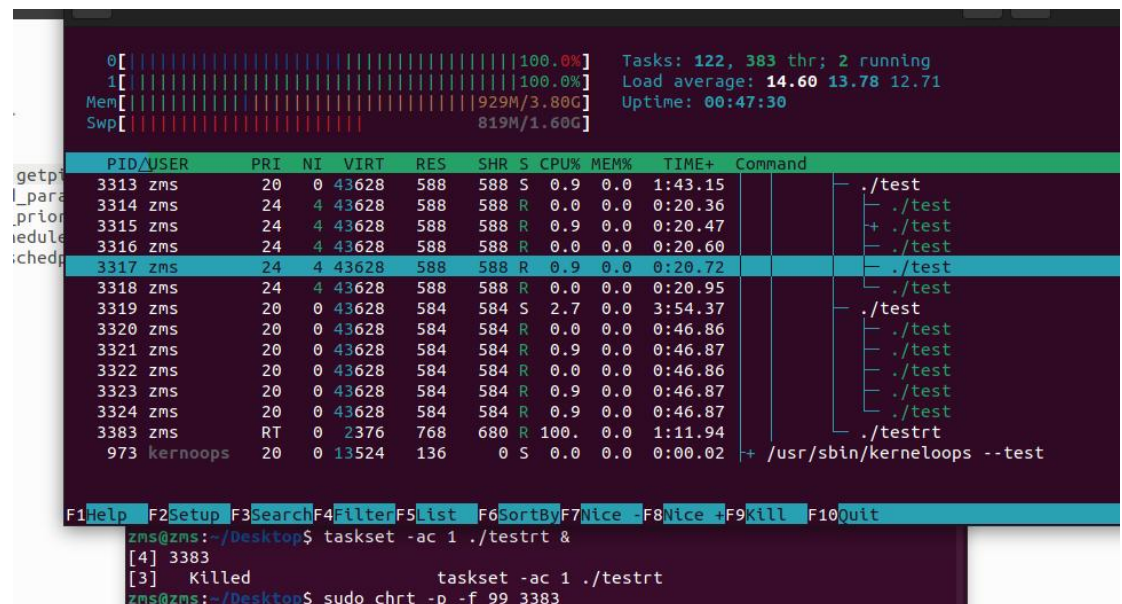


```
0[|||||100.0%] Tasks: 120, 276 thr; 2 running
1[|||||100.0%] Load average: 9.24 3.34 1.23
Mem[|||||1.00G/3.80G] Uptime: 00:02:42
Swp[|||||4.54M/1.60G]

  PID/USER     PRI  NI  VIRT  RES  SHR  S  CPU%  MEM%  TIME+  Command
  ----
 2131 zms       20   0 43628  792  696  S  29.6  0.0  0:16.69  ./test
 2132 zms       24   4 43628  792  696  R  5.1  0.0  0:03.05  ./test
 2133 zms       24   4 43628  792  696  R  5.1  0.0  0:03.19  ./test
 2134 zms       24   4 43628  792  696  R  7.1  0.0  0:03.32  ./test
 2135 zms       24   4 43628  792  696  R  7.1  0.0  0:03.45  ./test
 2136 zms       24   4 43628  792  696  R  6.1  0.0  0:03.64  ./test
 2145 zms       20   0 43628  732  632  S  71.4  0.0  0:28.69  ./test
 2146 zms       20   0 43628  732  632  R  14.3  0.0  0:05.72  ./test
 2147 zms       20   0 43628  732  632  R  14.3  0.0  0:05.73  ./test
 2148 zms       20   0 43628  732  632  R  14.3  0.0  0:05.73  ./test
 2149 zms       20   0 43628  732  632  R  14.3  0.0  0:05.73  ./test
 2150 zms       20   0 43628  732  632  R  14.3  0.0  0:05.73  ./test
 973 kernoops  20   0 13524  452    0  S  0.0  0.0  0:00.00  /usr/sbin/kernoops --test

F1Help F2Setup zms@zms: ~/Desktop
testrt.c
zms@zms:~/Desktop$ taskset -ac 1 ./test &
[1] 2131
zms@zms:~/Desktop$ taskset -ac 1 ./test &
[2] 2145
```

Task1.2 用 `sudo chrt -p -r 99 [pid]` 设置为实时进程



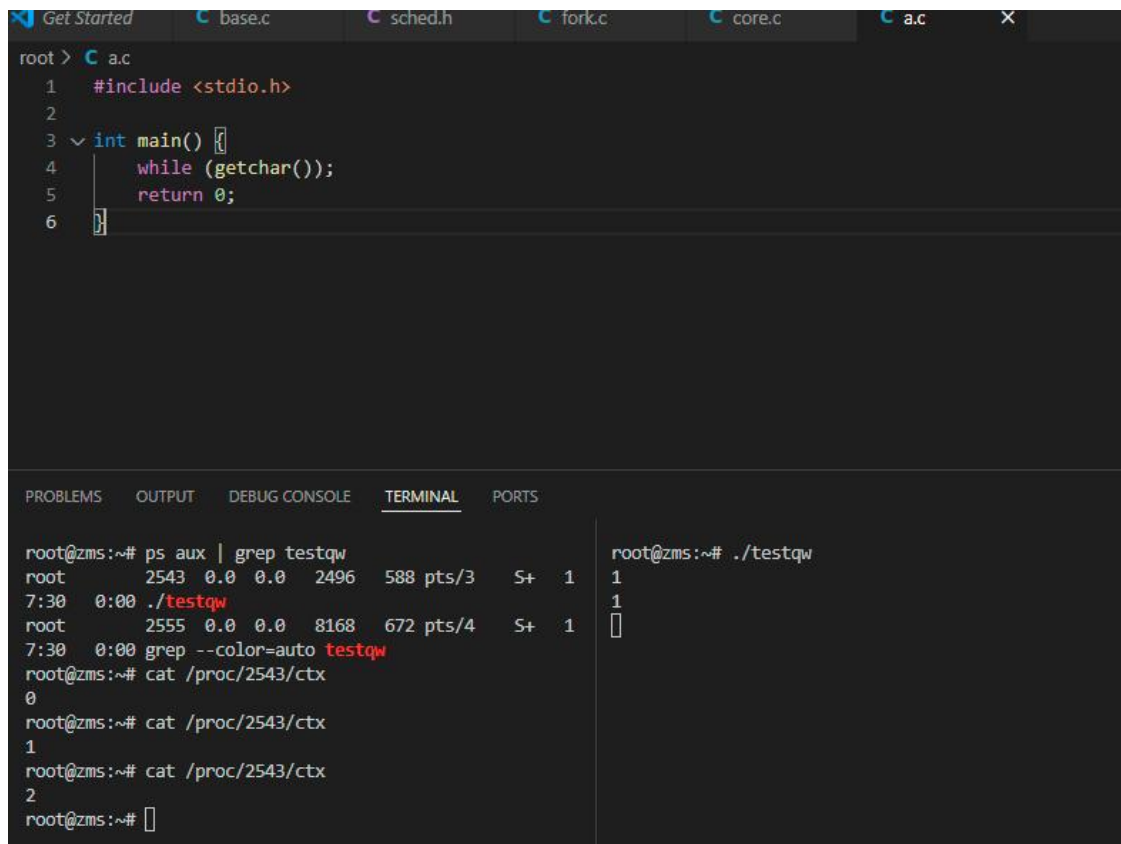
```
0[|||||100.0%] Tasks: 122, 383 thr; 2 running
1[|||||100.0%] Load average: 14.60 13.78 12.71
Mem[|||||929M/3.80G] Uptime: 00:47:30
Swp[|||||819M/1.60G]

  PID/USER     PRI  NI  VIRT  RES  SHR  S  CPU%  MEM%  TIME+  Command
  ----
 3313 zms       20   0 43628  588  588  S  0.9  0.0  1:43.15  ./test
 3314 zms       24   4 43628  588  588  R  0.0  0.0  0:20.36  ./test
 3315 zms       24   4 43628  588  588  R  0.9  0.0  0:20.47  ./test
 3316 zms       24   4 43628  588  588  R  0.0  0.0  0:20.60  ./test
 3317 zms       24   4 43628  588  588  R  0.9  0.0  0:20.72  ./test
 3318 zms       24   4 43628  588  588  R  0.0  0.0  0:20.95  ./test
 3319 zms       20   0 43628  584  584  S  2.7  0.0  3:54.37  ./test
 3320 zms       20   0 43628  584  584  R  0.0  0.0  0:46.86  ./test
 3321 zms       20   0 43628  584  584  R  0.9  0.0  0:46.87  ./test
 3322 zms       20   0 43628  584  584  R  0.0  0.0  0:46.86  ./test
 3323 zms       20   0 43628  584  584  R  0.9  0.0  0:46.87  ./test
 3324 zms       20   0 43628  584  584  R  0.9  0.0  0:46.87  ./test
 3383 zms       RT   0 2376  768  680  R 100.0  0.0  1:11.94  ./testrt
 973 kernoops  20   0 13524  136    0  S  0.0  0.0  0:00.02  /usr/sbin/kernoops --test

F1Help F2Setup F3Search F4Filter F5List F6SortBy F7Nice F8Nice + F9Kill F10Quit
zms@zms:~/Desktop$ taskset -ac 1 ./testrt &
[4] 3383
[3] Killed taskset -ac 1 ./testrt
zms@zms:~/Desktop$ sudo chrt -p -f 99 3383
```

图中 `cpu0` 为满是因为干扰进程

Task2 效果如图



The screenshot shows a code editor with several files: `Get Started`, `base.c`, `sched.h`, `fork.c`, `core.c`, and `a.c`. The `a.c` file is open, showing the following code:

```
1 #include <stdio.h>
2
3 int main() {
4     while (getchar());
5     return 0;
6 }
```

Below the code editor is a terminal window with the following output:

```
root@zms:~# ps aux | grep testqw
root      2543  0.0  0.0  2496   588 pts/3    S+   1
7:30    0:00  ./testqw
root      2555  0.0  0.0  8168   672 pts/4    S+   1
7:30    0:00  grep --color=auto testqw
root@zms:~# cat /proc/2543/ctx
0
root@zms:~# cat /proc/2543/ctx
1
root@zms:~# cat /proc/2543/ctx
2
root@zms:~#
```

On the right side of the terminal, the output of `./testqw` is shown:

```
root@zms:~# ./testqw
1
1
[]
```

实验过程

在 `task_struct` 结构体中添加数据成员变量 `int ctx`

在 `fork.c` 中查找 `task_struct` 引用，发现 `copy_process` 函数，根据函数名猜测与创建函数相关，添加 `p->ctx=0`

在 `core.c` 中查找调度相关，发现 `wake_up_process` 由多层调用关系又发现 `sched_update_worker` 两者均尝试后选择 `sched_update_worker` 添加 `tsk->ctx++`

在 `Base.c` 中添加注册文件函数，仿照 `array.c` 中 `proc_tid_stat` 实现