CSGE602055 Operating Systems
CSF2600505 Sistem Operasi
Week 05: Virtual Memory

Rahmat M. Samik-Ibrahim (ed.)

University of Indonesia

https://os.vlsm.org/
Always check for the latest revision!

REV236 13-Sep-2020

Operating Systems 2020-2 – (A, I, M) **from HOME** A/M [Tu 10-12, ZOOM] — I [Tu 08-10, ZOOM]

Week	Schedule	Topic	OSC10
Week 00	15 Sep - 21 Sep 2020	Overview 1, Virtualization & Scripting	Ch. 1, 2, 18.
Week 01	22 Sep - 28 Sep 2020	Overview 2, Virtualization & Scripting	Ch. 1, 2, 18.
Week 02	29 Sep - 05 Oct 2020	Security, Protection, Privacy, & C-language	Ch. 16, 17
Week 03	06 Oct - 12 Oct 2020	File System & FUSE	Ch. 13, 14, 15
Week 04	13 Oct - 19 Oct 2020	Addressing, Shared Lib, & Pointer	Ch. 9
Week 05	20 Oct - 26 Oct 2020	Virtual Memory	Ch. 10
Week 06	27 Oct - 16 Nov 2020 29 Oct 2020	Concurrency: Processes & Threads Maulid Nabi	Ch. 3, 4
Week 07	17 Nov - 23 Nov 2020	Synchronization & Deadlock	Ch. 6, 7, 8
Week 08	24 Nov - 30 Nov 2020	Scheduling + W06/W07	Ch. 5
Week 09	01 Dec - 07 Dec 2020	Storage, Firmware, Bootldr, & Systemd	Ch. 11
Week 10	08 Dec - 16 Dec 2020	I/O & Programming	Ch. 12
	09 Dec 2020	Pil Kada	

STARTING POINT — https://os.vlsm.org/

- **Text Book** — Any recent/decent OS book. Eg. (**OSC10**) Silberschatz et. al.: **Operating System Concepts**, 10^{th} Edition, 2018. See also http://codex.cs.yale.edu/avi/os-book/OS10/.
- **Resources**
  - **Extra Scele from Home** — https://scele.cs.ui.ac.id/course/view.php?id=3020.
  - **Download Slides and Demos from GitHub.com** https://github.com/UI-FASILKOM-OS/SistemOperasi/
  - **Problems** — https://rms46.vlsm.org/2/: 195.pdf (W00), 196.pdf (W01), 197.pdf (W02), 198.pdf (W03), 199.pdf (W04), 200.pdf (W05), 201.pdf (W06), 202.pdf (W07), 203.pdf (W08), 204.pdf (W09), 205.pdf (W10).
- **Build your own Virtual Guest** https://osp4diss.vlsm.org/

Week 05: Memory

Week 05 Virtual Memory: Topics[1]

- Review of physical memory and memory management hardware
- Virtual Memory
- Caching
- Memory Allocation
- Memory Performance
- Working sets and thrashing

[1] Source: ACM IEEE CS Curricula 2013

Week 05 Virtual Memory: Learning Outcomes[1]

- Explain memory hierarchy and cost-performance trade-offs. [Familiarity]
- Summarize the principles of virtual memory as applied to caching and paging. [Familiarity]
- Describe the reason for and use of cache memory (performance and proximity, different dimension of how caches complicate isolation and VM abstraction). [Familiarity]
- Defend the different ways of allocating memory to tasks, citing the relative merits of each. [Assessment]
- Evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed. [Assessment]
- Discuss the concept of thrashing, both in terms of the reasons it occurs and the techniques used to recognize and manage the problem. [Familiarity]

[1] Source: ACM IEEE CS Curricula 2013

Virtual Memory

- Reference: (OSC10-ch10 demo-w05)
- Virtual Memory: Separation Logical from Physical.
- Virtual Address Space: logical view.
- Demand Paging
- Page Flags: Valid / Invalid
- Page Fault
- Demand Paging Performance
- Copy On Write (COW)
- Page Replacement Algorithm
  - Reference String
  - First-In-First-Out (FIFO)
  - Belady Anomaly
  - Optimal Algorithm
  - Least Recently Used (LRU)
  - LRU Implementation
  - Lease Frequently Used (LFU)
  - Most Frequently Used (MFU)

Allocation Algorothm

- Page-Buffering Algorithms
- Allocation of Frames
- Fixed Allocation
- Priority Allocation
- Global vs. Local Allocation
- Non-Uniform Memory Access (NUMA)
- Thrashing
- Working-Set Model
- Shared Memory via Memory-Mapped I/O
- Kernel
  - Buddy System Allocator
  - Slab Allocator

TOP

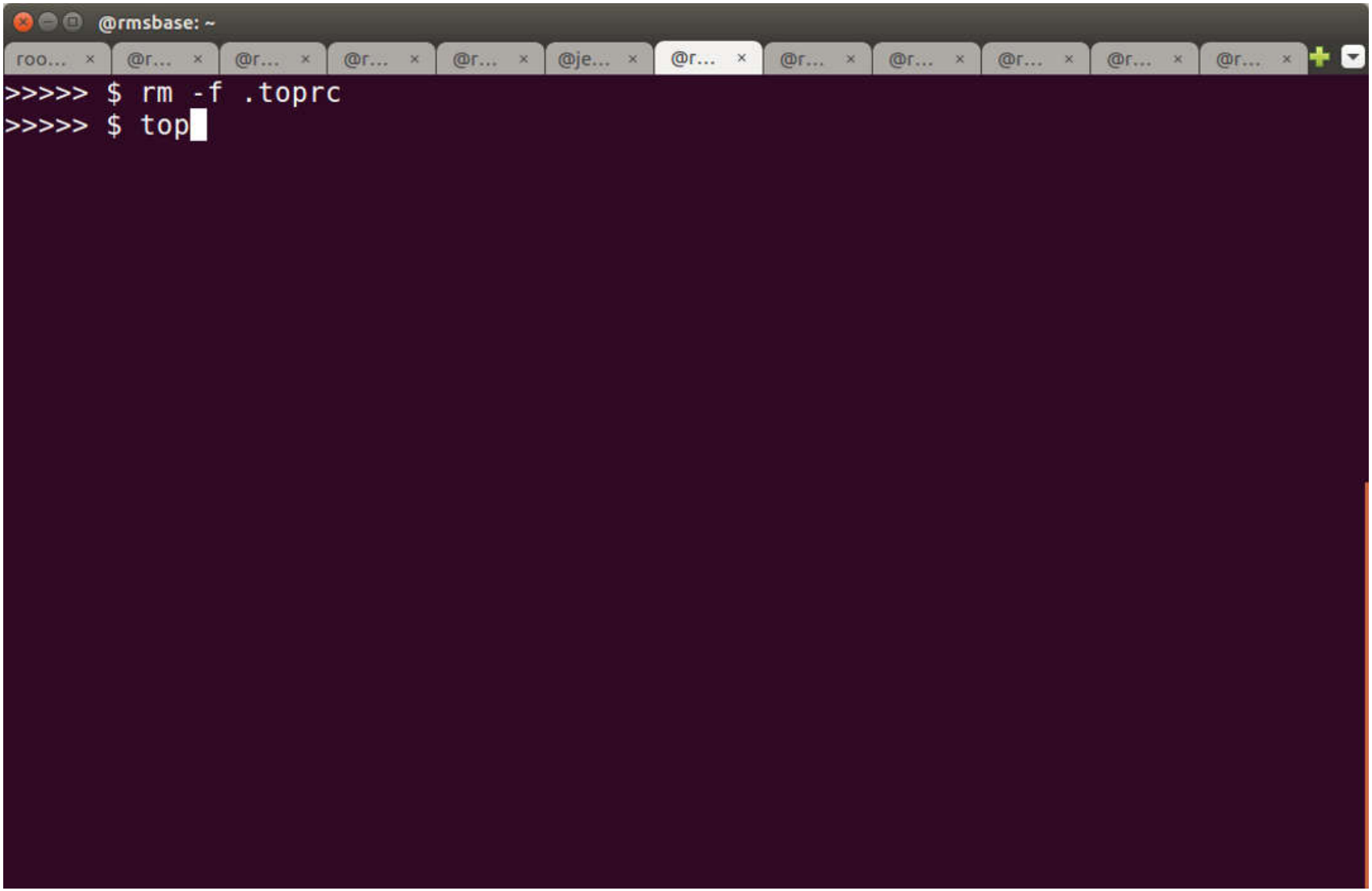

Figure: top

TOP (2)

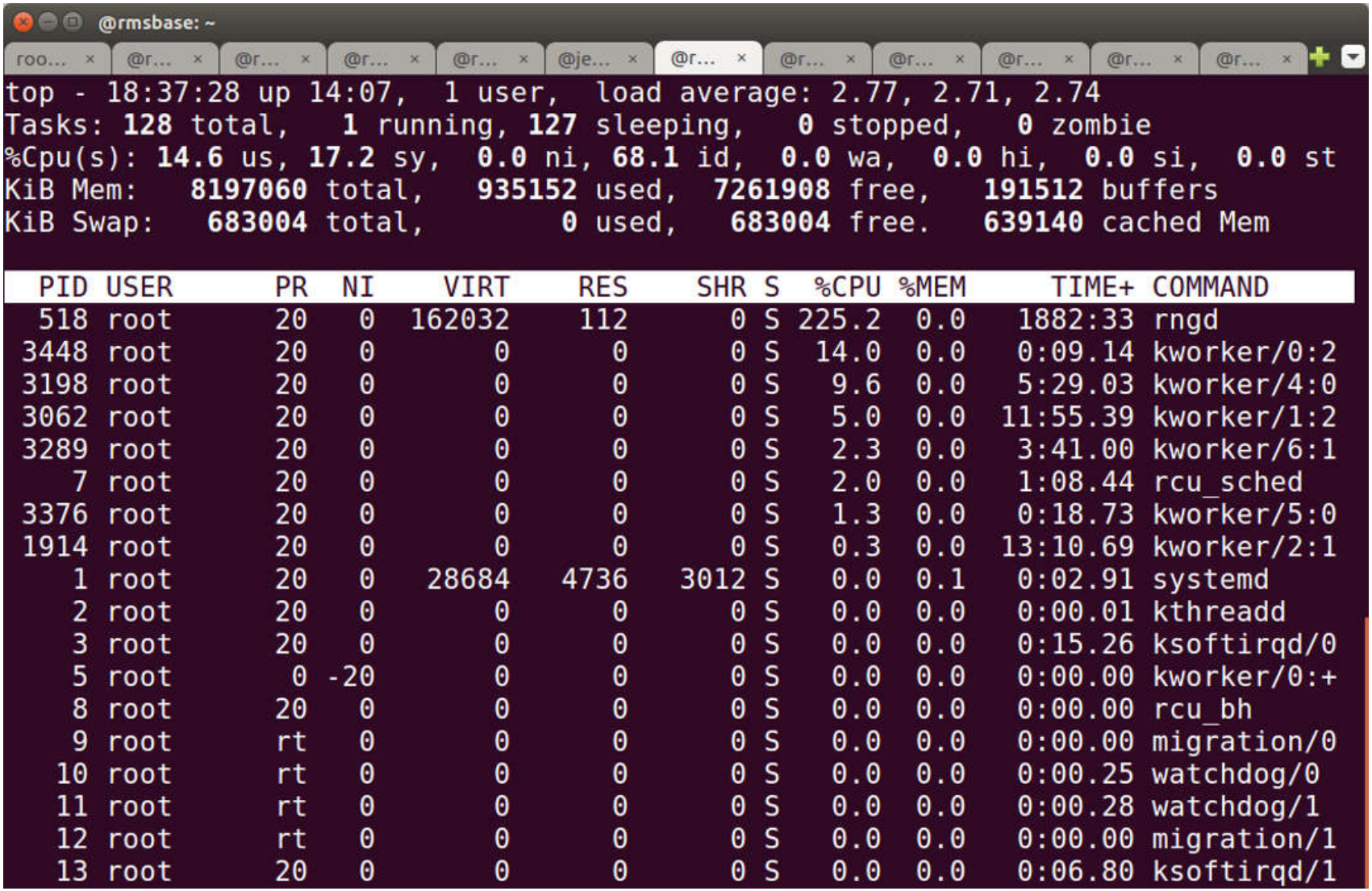

Figure: "h" = help

TOP (3)

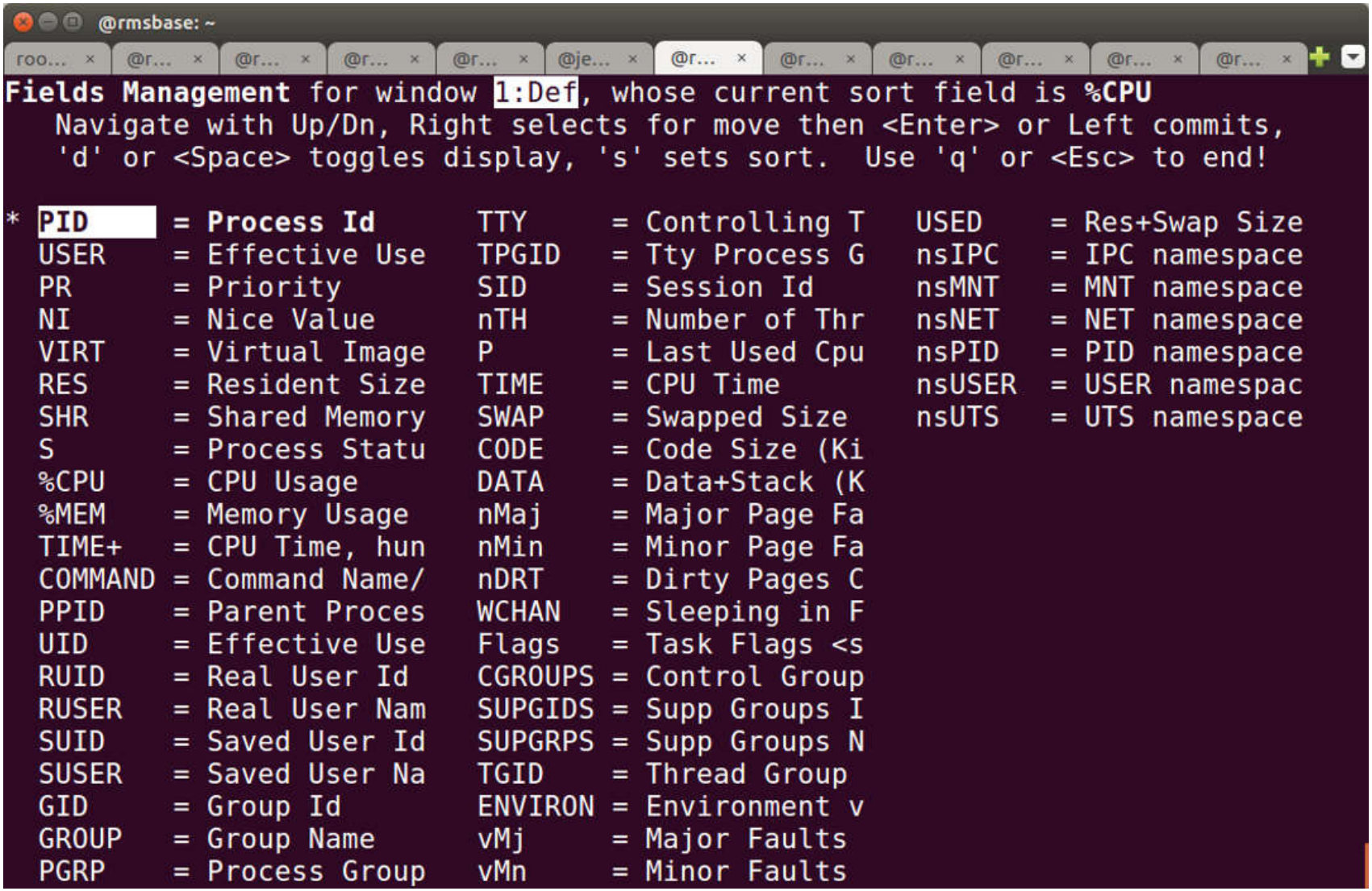

Figure: Moving Fields: "f"

TOP (4)

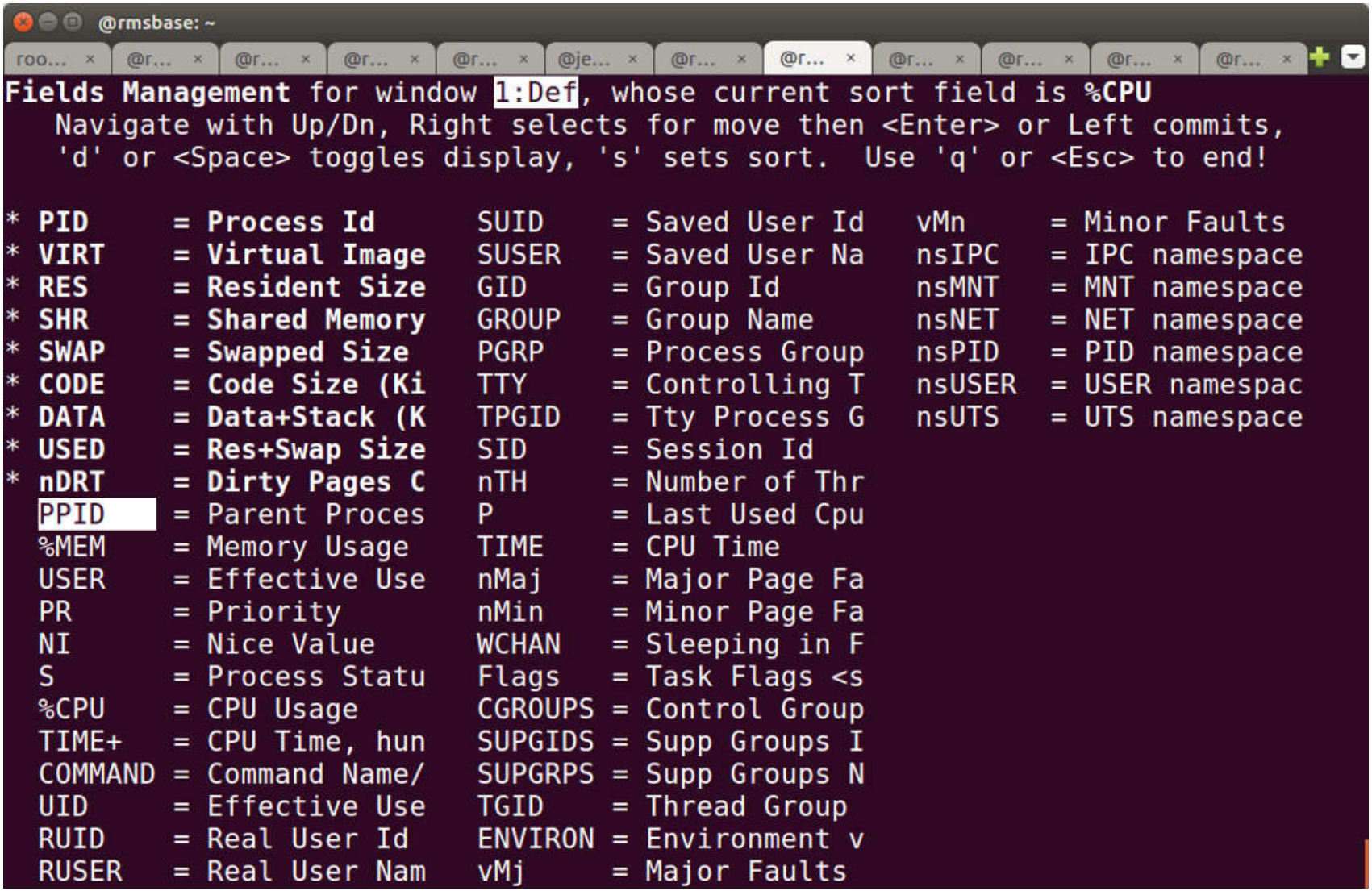

Figure: Moving Fields

TOP (5)

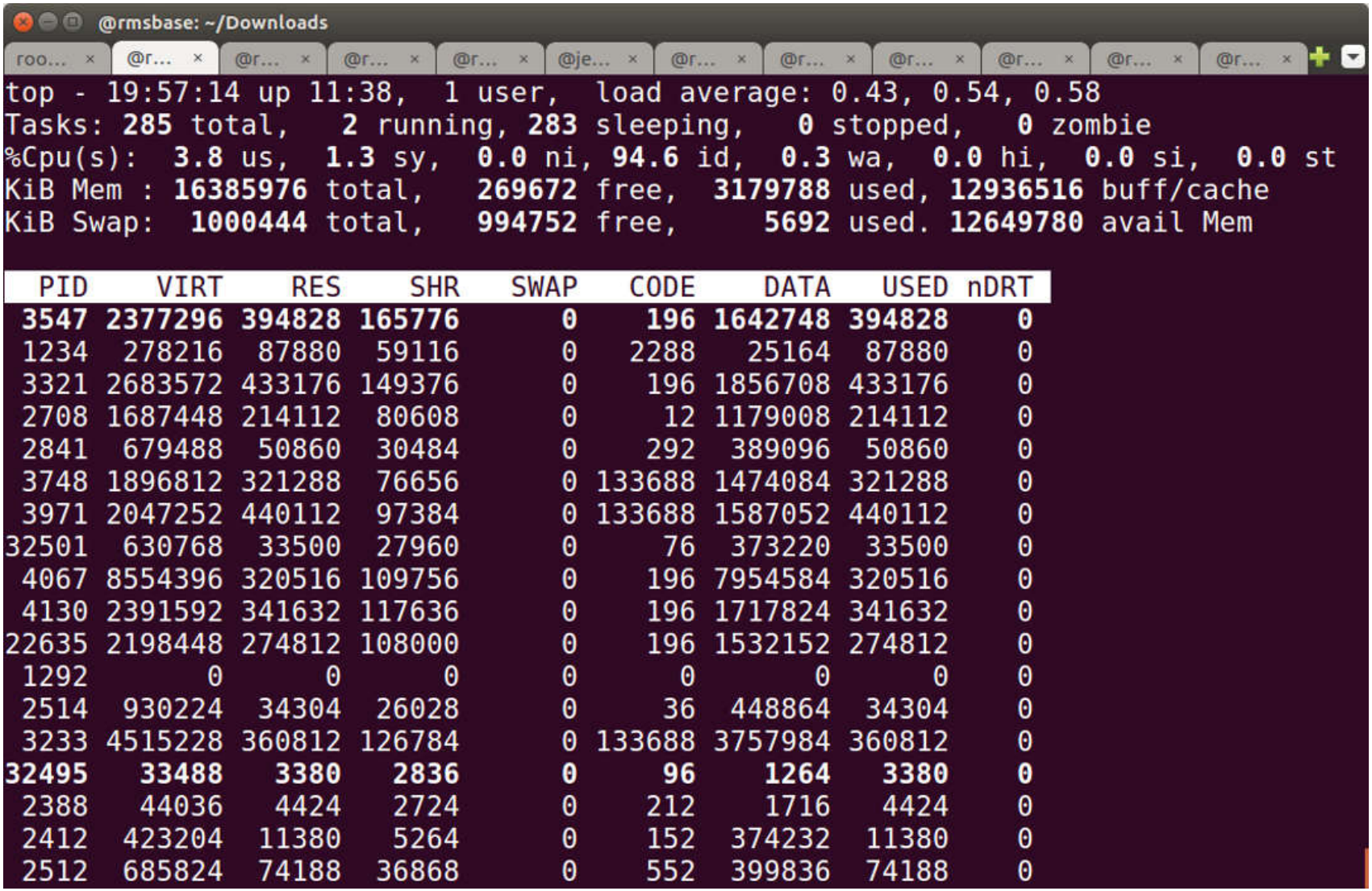

Figure: Write Configuration .toprc: "W"

06-memory

```
/* Copyright (C) 2016-2018 Rahmat M. Samik-Ibrahim
 * https://rahmatm.samik-ibrahim.vlsm.org/
 * This program is free script/software. This program is distributed in the
 * hope that it will be useful, but WITHOUT ANY WARRANTY; without even the
 * implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
 * REV04 Mon Mar 12 17:33:30 WIB 2018
 * START Mon Oct  3 09:26:51 WIB 2016
 */
#define MSIZE0 0x10000
#define MSIZE1 0x10008
#define MSIZE2 0x10009
#define MSIZE3 0x1000A
#define MSIZE4 0x20978
#define MSIZE5 0x20979
#define MSIZE6 0x2097A
#define MSIZE7 0xF0000
#define MSIZE8 0x10000
#define MSIZE9 0x1000
#define LINE   75
#define MAXSTR 80
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>

void printLine(int line) {
   while(line-- > 0) putchar('x');
   putchar('\n');
   fflush(NULL);
}
```

06-memory (2)

```
void main (void) {
   int   msize[] = {MSIZE0, MSIZE1, MSIZE2, MSIZE3, MSIZE4,
                    MSIZE5, MSIZE6, MSIZE7, MSIZE8, MSIZE9};
   int   ii, jj;
   int   myPID   = (int) getpid();
   char  strSYS1[MAXSTR], strOUT[MAXSTR];
   char* chrStr  = strSYS1;
   char* chrPTR;

   printLine(LINE);
   sprintf(strSYS1, "top -b -n 1 -p%d | tail -5", myPID);
   system (strSYS1);
   sprintf(strSYS1, "top -b -n 1 -p%d | tail -1", myPID);
   for (ii=0; ii< (sizeof(msize)/sizeof(int)); ii++){
      chrStr = malloc(msize[ii]);
      fgets(strOUT, sizeof(strOUT)-1, popen(strSYS1, "r"));
      strOUT[(int) strlen(strOUT)-1]='\0';
      printf("%s [%X]\n", strOUT, msize[ii]);
      free(chrStr);
   }
   for (ii=0; ii< (sizeof(msize)/sizeof(int)); ii++){
      chrPTR = chrStr = malloc(msize[ii]);
      for (jj=0;jj<msize[ii];jj++)
         *chrPTR++='x';
      fgets(strOUT, sizeof(strOUT)-1, popen(strSYS1, "r"));
      strOUT[(int) strlen(strOUT)-1]='\0';
      printf("%s [%X]\n", strOUT, msize[ii]);
      free(chrStr);
   }
}
```

06-memory (2)

```
>>>>> $ ./06-memory
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
KiB Mem:   8197060 total,   957928 used,  7239132 free,   192520 buffers
KiB Swap:   683004 total,        0 used,   683004 free.   660108 cached
    Mem

  PID    VIRT    RES    SHR   SWAP   CODE    DATA   USED nDRT
 4362    4172    640    564      0      4     320    640    0
 4362    4172    640    564      0      4     320    640    0 [10000]
 4362    4172    640    564      0      4     320    640    0 [10008]
 4362    4308    640    564      0      4     456    640    0 [10009]
 4362    4244   1176   1068      0      4     392   1176    0 [1000A]
 4362    4244   1176   1068      0      4     392   1176    0 [20978]
 4362    4376   1176   1068      0      4     524   1176    0 [20979]
 4362    4376   1192   1068      0      4     524   1192    0 [2097A]
 4362    5340   1192   1068      0      4    1488   1192    0 [F0000]
 4362    4376   1200   1068      0      4     524   1200    0 [10000]
 4362    4376   1200   1068      0      4     524   1200    0 [1000]
```

06-memory (3)

```
 4362    4376    1200    1068    0    4    524    1200    0 [1000]
 4362    4376    1200    1068    0    4    524    1200    0 [10000]
 4362    4376    1276    1068    0    4    524    1276    0 [10008]
 4362    4376    1276    1068    0    4    524    1276    0 [10009]
 4362    4376    1284    1068    0    4    524    1284    0 [1000A]
 4362    4376    1284    1068    0    4    524    1284    0 [20978]
 4362    4376    1352    1068    0    4    524    1352    0 [20979]
 4362    4376    1352    1068    0    4    524    1352    0 [2097A]
 4362    5340    2144    1068    0    4   1488    2144    0 [F0000]
 4362    5340    2324    1068    0    4   1488    2324    0 [10000]
 4362    5340    2324    1068    0    4   1488    2324    0 [1000]
>>>>> $
```

The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
- This is the end of the presentation.