Southern Connecticut
State University

CSC424 System Administration

Instructor: Dr. Hao Wu

Week 6 Bash Scripting - 1

Outline

- Filename Matching
- Regular Expression
- Linux Text Processing Tools
- Variables
- Arithmetic
- Control Flow
- Loops
- Function

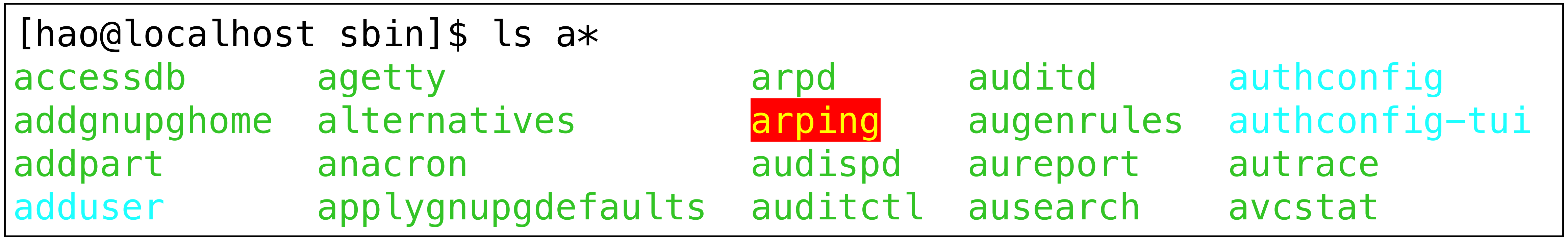
Filename Matching

- Wildcards (meta characters) are symbols or special characters that represent other characters. You can use them with any command such as ls, cp, mv, rm to perform file operations matching a given criteria.
- Three main wildcards in Linux:
  - * : matches one or more occurrences of any character, including no character
  - ? : represents or matches a single occurrence of any character
  - [] : matches any occurrence of character enclosed in the square bracket. It is possible to use different types of characters (alphanumeric characters): numbers, letters, other special characters etc.)

Zero or multiple characters matching (*)

1. If we want to list all the files start with 'a':

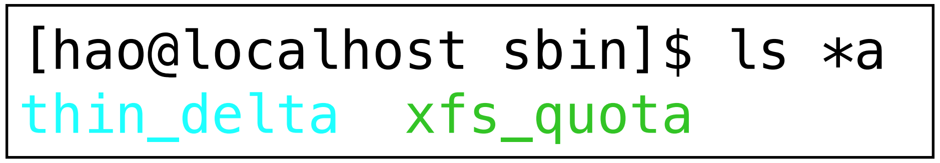
ls a*



```
[hao@localhost sbin]$ ls a*
accessdb      agetty               arpd      auditd      authconfig
addgnupghome  alternatives         arping    augenrules  authconfig-tui
addpart       anacron              audispd   aureport    autrace
adduser       applygnupgdefaults   auditctl  ausearch    avcstat
```

2. If we want to list all the files end with 'a'

ls *a



```
[hao@localhost sbin]$ ls *a
thin_delta  xfs_quota
```

Zero or multiple characters matching

1. If we want to list all the files start with 'a' and end with 't':

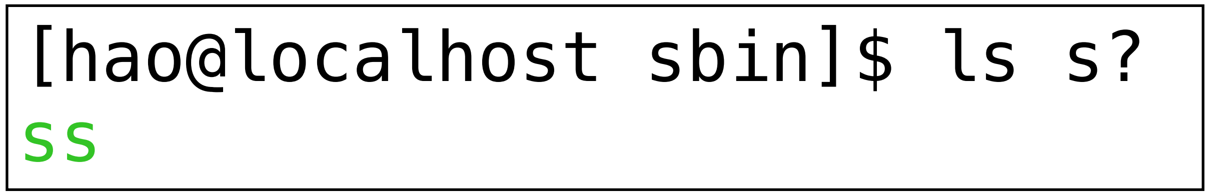
ls a*t

```
[hao@localhost sbin]$ ls a*t
addpart   aureport   avcstat
```

Single character matching (?)

1. If we want to list a file whose name contains 2 characters and starts with 's'

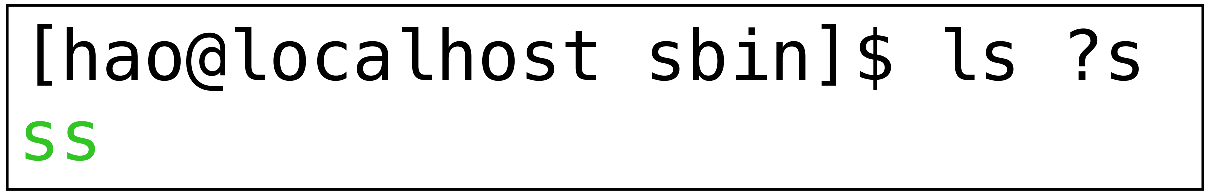
ls s?



```
[hao@localhost sbin]$ ls s?
ss
```

2. If we want to list a file whose name contains 2 characters and ends with 's'

ls ?s

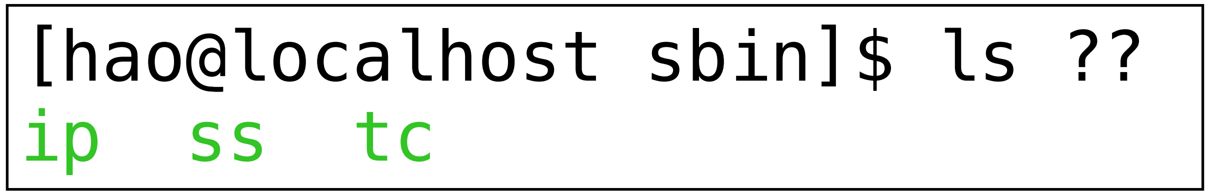


```
[hao@localhost sbin]$ ls ?s
ss
```

Single character matching (?)

1. If we want to list a file whose name contains 2 characters

ls ??

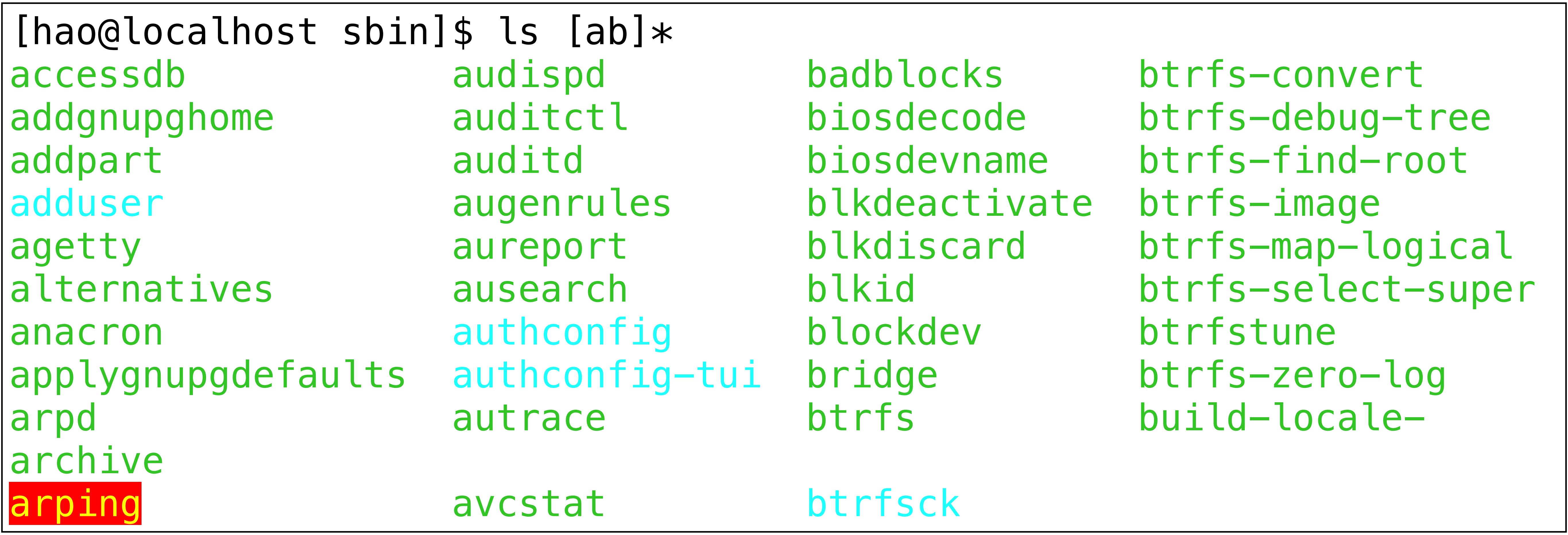


```
[hao@localhost sbin]$ ls ??
ip  ss  tc
```

Range of character matching ([])

1. If we want to list files whose names start with 'a' or 'b'

ls [ab]*

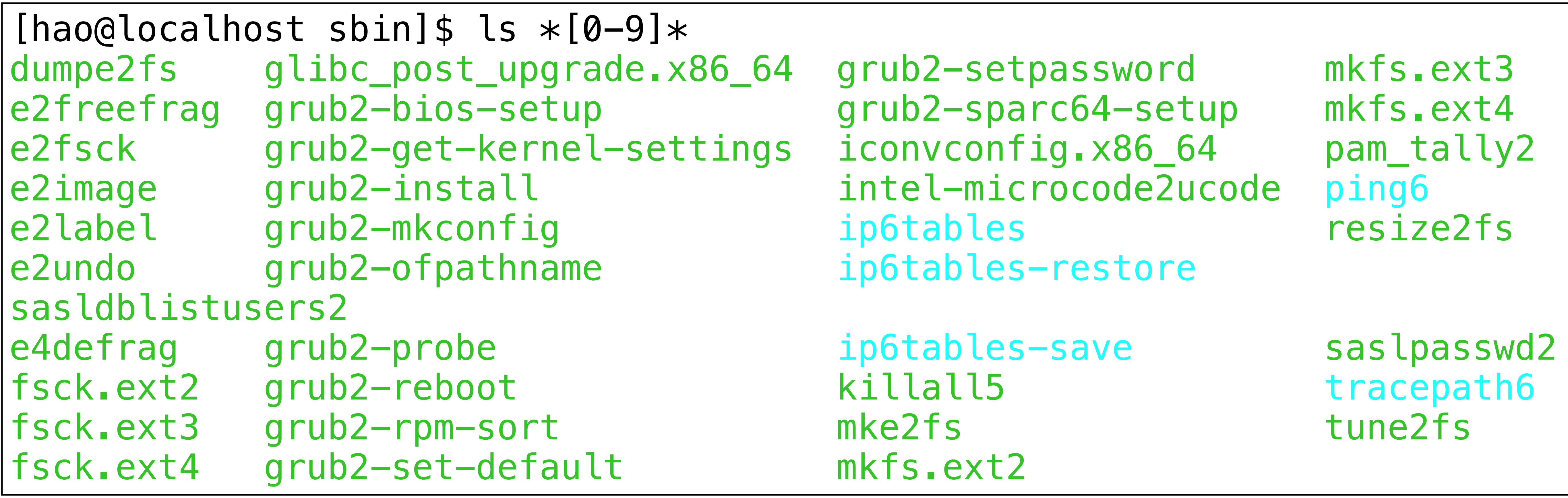


```
[hao@localhost sbin]$ ls [ab]*
accessdb            audispd         badblocks      btrfs-convert
addgnupghome        auditctl        biosdecode     btrfs-debug-tree
addpart             auditd          biosdevname    btrfs-find-root
adduser             augenrules      blkdeactivate  btrfs-image
agetty              aureport        blkdiscard     btrfs-map-logical
alternatives        ausearch        blkid          btrfs-select-super
anacron             authconfig      blockdev       btrfstune
applygnupgdefaults  authconfig-tui  bridge         btrfs-zero-log
arpd                autrace         btrfs          build-locale-
archive
arping              avcstat         btrfsck
```

Range of character matching ([])

1. If we want to list files whose names contain numbers

   ls *[0-9]*

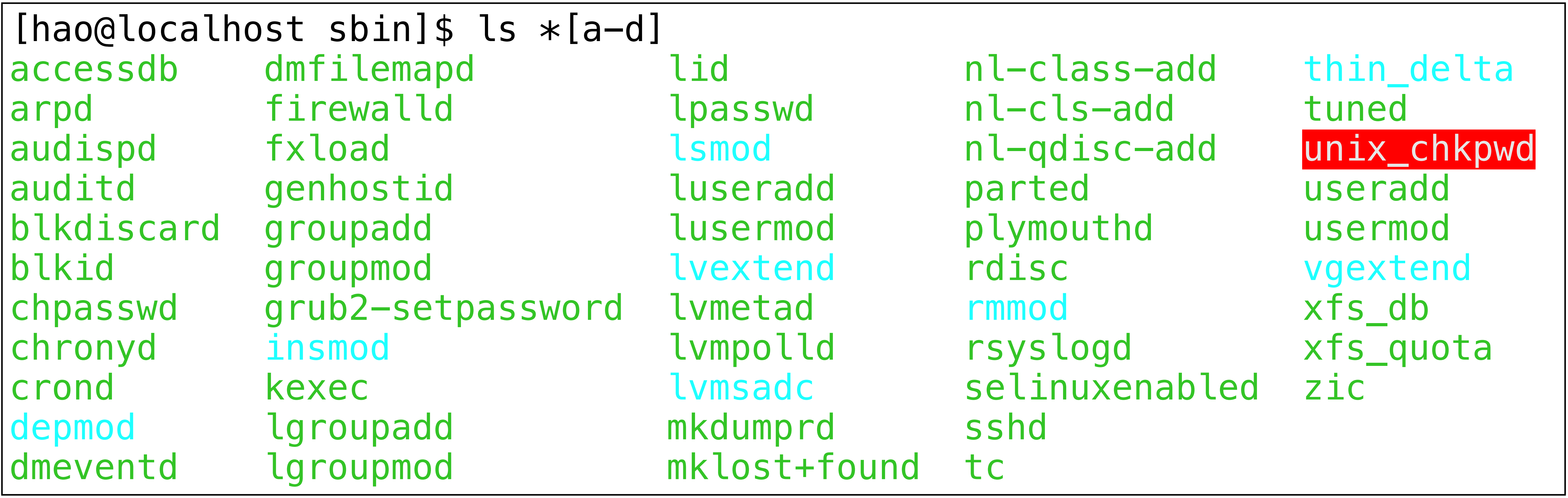


```
[hao@localhost sbin]$ ls *[0-9]*
dumpe2fs    glibc_post_upgrade.x86_64  grub2-setpassword      mkfs.ext3
e2freefrag  grub2-bios-setup           grub2-sparc64-setup    mkfs.ext4
e2fsck      grub2-get-kernel-settings  iconvconfig.x86_64     pam_tally2
e2image     grub2-install              intel-microcode2ucode  ping6
e2label     grub2-mkconfig             ip6tables              resize2fs
e2undo      grub2-ofpathname           ip6tables-restore
sasldblistusers2
e4defrag    grub2-probe                ip6tables-save         saslpasswd2
fsck.ext2   grub2-reboot               killall5               tracepath6
fsck.ext3   grub2-rpm-sort             mke2fs                 tune2fs
fsck.ext4   grub2-set-default          mkfs.ext2
```

Range of character matching ([])

1. If we want to list files whose names end with 'a-d'

ls *[a-d]



```
[hao@localhost sbin]$ ls *[a-d]
accessdb    dmfilemapd         lid           nl-class-add    thin_delta
arpd        firewalld          lpasswd       nl-cls-add      tuned
audispd     fxload             lsmod         nl-qdisc-add    unix_chkpwd
auditd      genhostid          luseradd      parted          useradd
blkdiscard  groupadd           lusermod      plymouthd       usermod
blkid       groupmod           lvextend      rdisc           vgextend
chpasswd    grub2-setpassword  lvmetad       rmmod           xfs_db
chronyd     insmod             lvmpolld      rsyslogd        xfs_quota
crond       kexec              lvmsadc       selinuxenabled  zic
depmod      lgroupadd          mkdumprd      sshd
dmeventd    lgroupmod          mklost+found  tc
```

Regular Expression

- Regular expressions are standardized patterns that parse and manipulate text. For example, the regular expression:

- I sent you a che(que|ck) for the gr[ae]y-colou?red alumni?um
- matches sentences that use either American or British specling conventions.

- Regular expressions are supported by most modern languages.
- Wildcard is not a form of regular expression!

Special characters in regular expression

Symbol	What it matches or does
.	Matches any character
[]	Matches any character from a given set
^	Matches the beginning of a line
$	Matches the end of a line
\w	Matches any "word" character
\s	Matches any whitespace character
\d	Matches any digits
|	Matches either the element to its left or right
?	Allows zero or one match of the preceding element
*	Allows zero, one, or many matches of the preceding elements
+	Allows one or more matches of the preceding element
{n}	Matches exactly n instances of the preceding elements
{min,}	Matches at least min instance of
{min, max}	Matches any number of instances from min to max

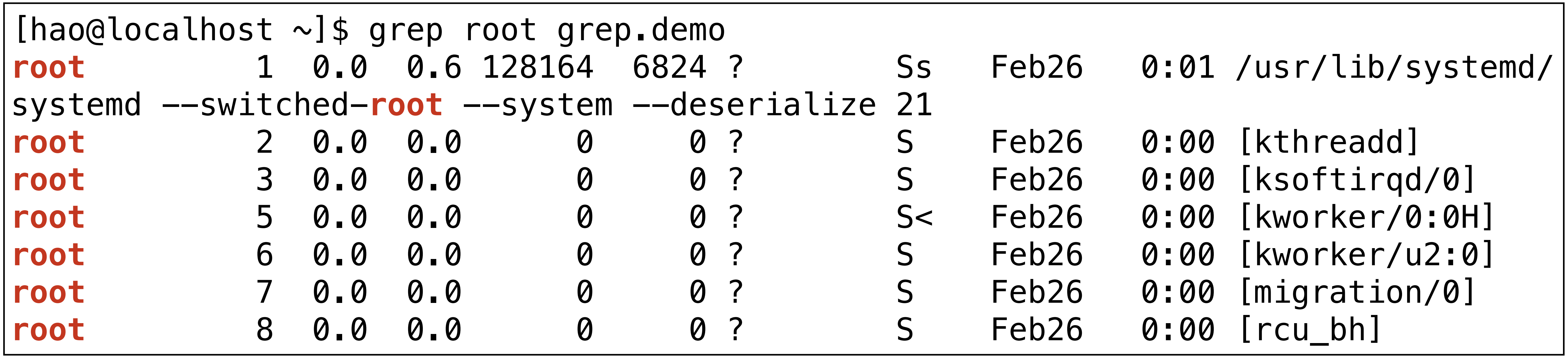
Text Processing Tools: grep

- grep: print lines matching a pattern
- Syntax:
- grep [option] pattern [file]
- Commonly used options:
  - i: ignore case distinctions in both the pattern and the input files.
  - n: Prefix each line of output with the line number within its input file
  - v: Invert the sense of matching, to select non-matching lines

- In basic regular expressions the meta-characters **?**, **+**, **{**, **|**, **(**, and **)** lose their special meaning; instead use the backslashed versions **\?**, **\+**, **\{**, **\|**, **\(**, and **\)**.

grep examples:

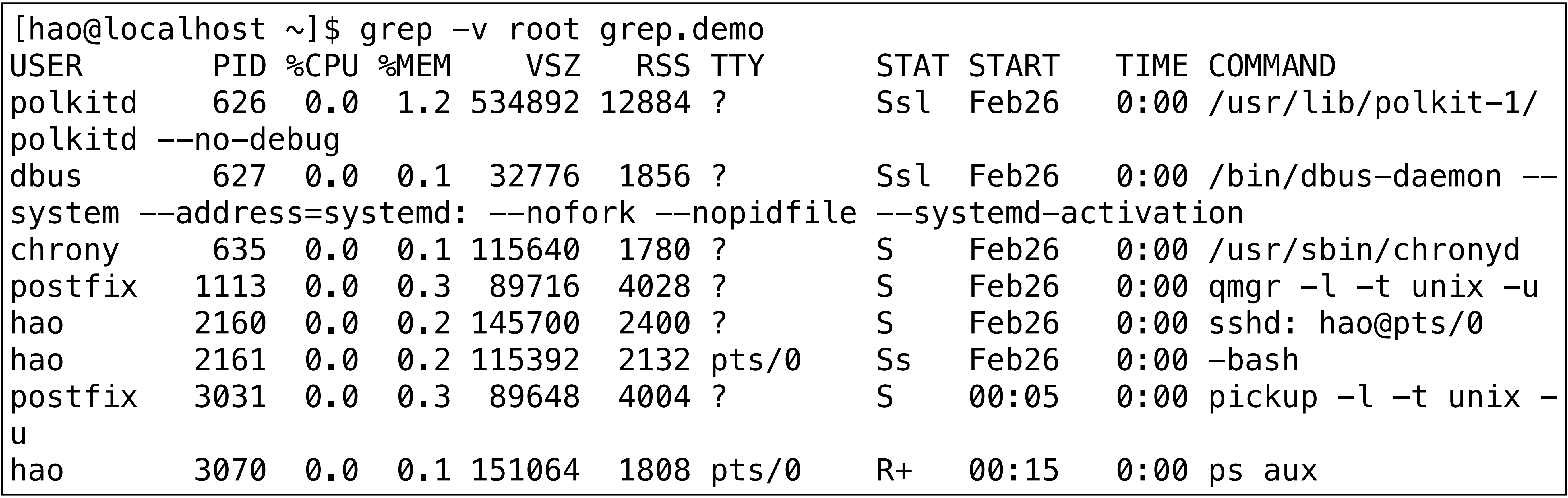
- We will first create a file for demonstration:
- ps aux > grep.demo
- If we want to find the lines contain 'root'
- grep root grep.demo



```
[hao@localhost ~]$ grep root grep.demo
root         1  0.0  0.6 128164  6824 ?        Ss   Feb26   0:01 /usr/lib/systemd/
systemd --switched-root --system --deserialize 21
root         2  0.0  0.0      0     0 ?        S    Feb26   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Feb26   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Feb26   0:00 [kworker/0:0H]
root         6  0.0  0.0      0     0 ?        S    Feb26   0:00 [kworker/u2:0]
root         7  0.0  0.0      0     0 ?        S    Feb26   0:00 [migration/0]
root         8  0.0  0.0      0     0 ?        S    Feb26   0:00 [rcu_bh]
```

grep examples:

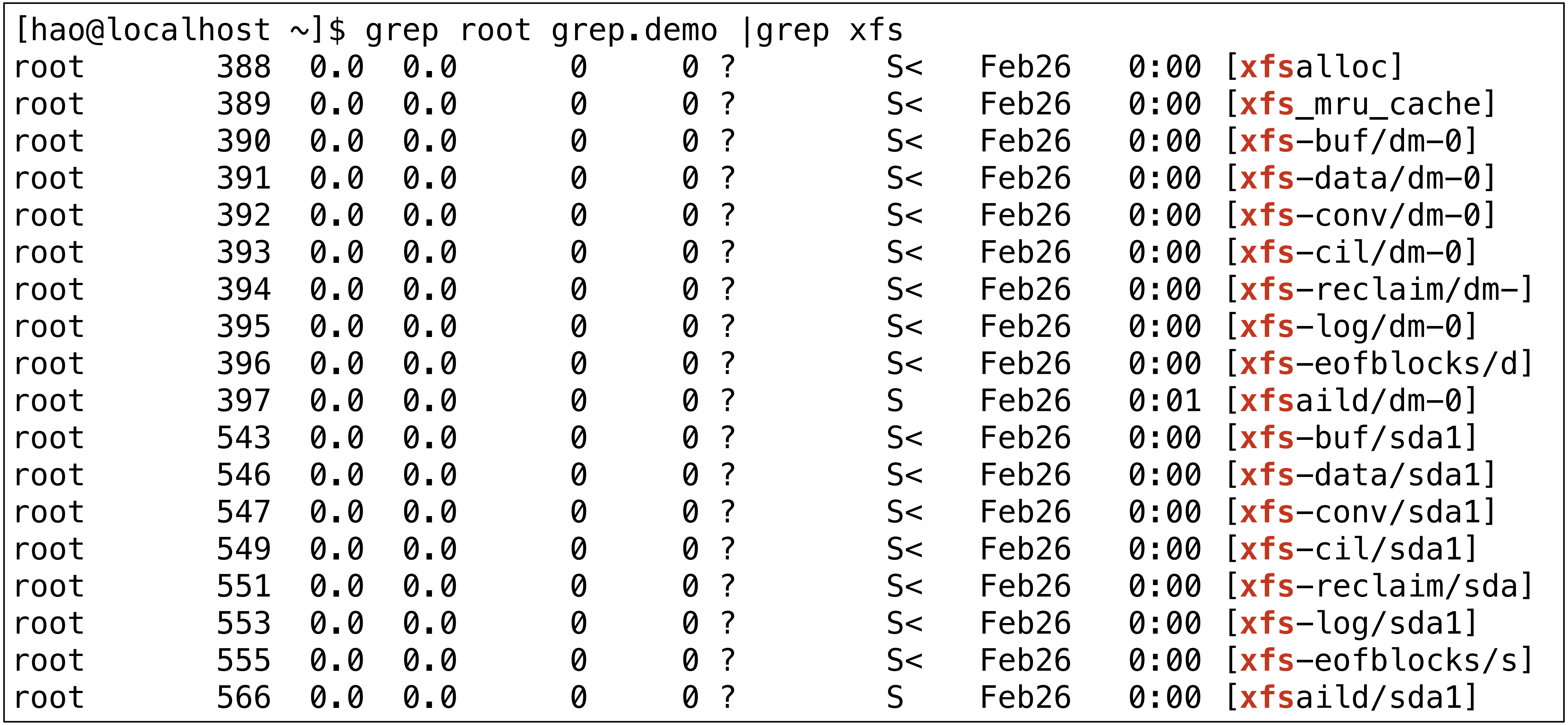
- If we want to find the lines do not contain 'root'
- grep -v root grep.demo



```
[hao@localhost ~]$ grep -v root grep.demo
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
polkitd     626  0.0  1.2 534892 12884 ?        Ssl  Feb26   0:00 /usr/lib/polkit-1/
polkitd --no-debug
dbus        627  0.0  0.1  32776  1856 ?        Ssl  Feb26   0:00 /bin/dbus-daemon --
system --address=systemd: --nofork --nopidfile --systemd-activation
chrony      635  0.0  0.1 115640  1780 ?        S    Feb26   0:00 /usr/sbin/chronyd
postfix    1113  0.0  0.3  89716  4028 ?        S    Feb26   0:00 qmgr -l -t unix -u
hao        2160  0.0  0.2 145700  2400 ?        S    Feb26   0:00 sshd: hao@pts/0
hao        2161  0.0  0.2 115392  2132 pts/0    Ss   Feb26   0:00 -bash
postfix    3031  0.0  0.3  89648  4004 ?        S    00:05   0:00 pickup -l -t unix -
u
hao        3070  0.0  0.1 151064  1808 pts/0    R+   00:15   0:00 ps aux
```

grep examples:

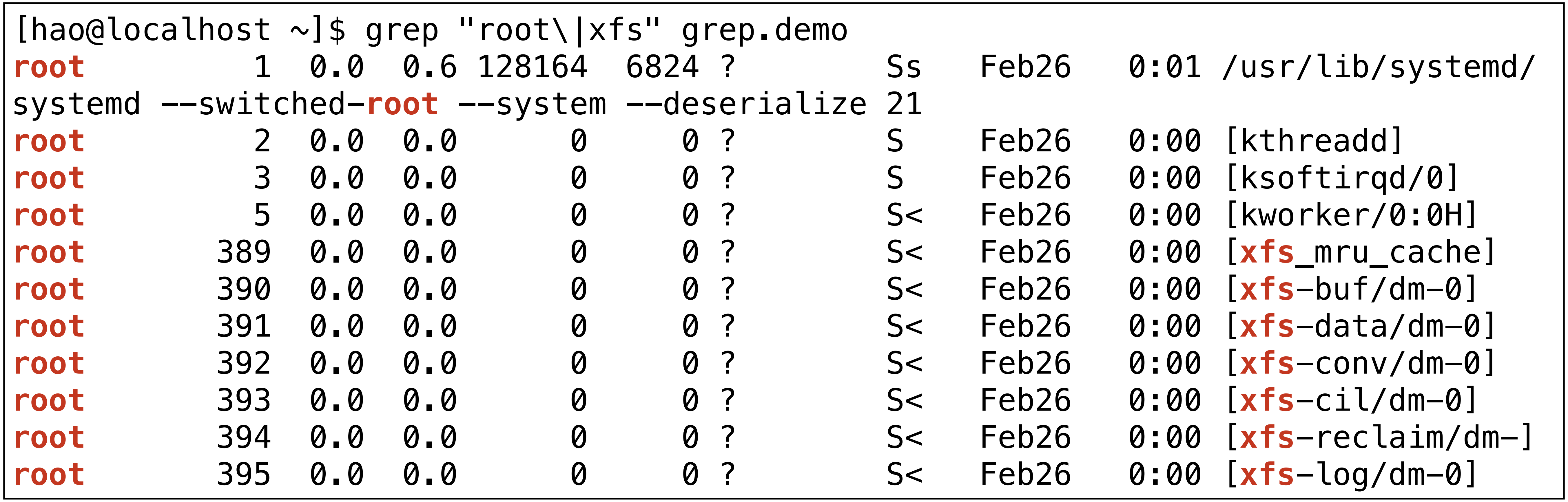
- If we want to find the lines contain 'root' and 'xfs'
- grep root grep.demo|grep xfs



```
[hao@localhost ~]$ grep root grep.demo |grep xfs
root       388  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfsalloc]
root       389  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs_mru_cache]
root       390  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-buf/dm-0]
root       391  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-data/dm-0]
root       392  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-conv/dm-0]
root       393  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-cil/dm-0]
root       394  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-reclaim/dm-]
root       395  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-log/dm-0]
root       396  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-eofblocks/d]
root       397  0.0  0.0      0     0 ?        S    Feb26   0:01 [xfsaild/dm-0]
root       543  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-buf/sda1]
root       546  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-data/sda1]
root       547  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-conv/sda1]
root       549  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-cil/sda1]
root       551  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-reclaim/sda]
root       553  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-log/sda1]
root       555  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-eofblocks/s]
root       566  0.0  0.0      0     0 ?        S    Feb26   0:00 [xfsaild/sda1]
```

grep examples:

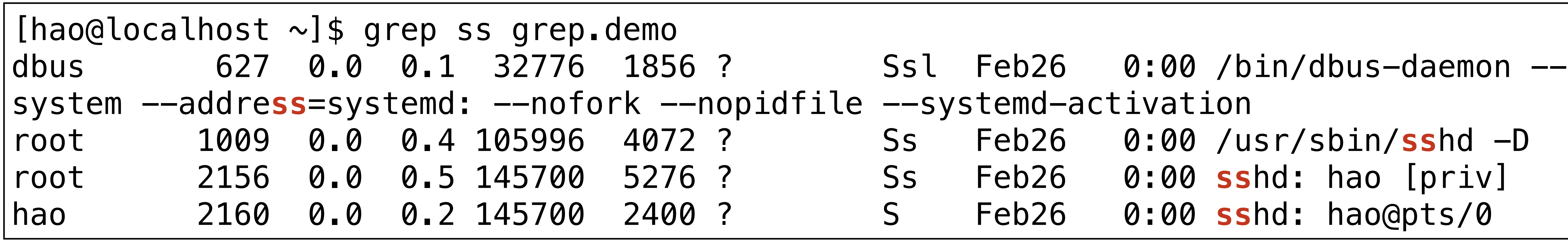
- If we want to find the lines contain 'root' or 'xfs'
- grep "root\|xfs" grep.demo



```
[hao@localhost ~]$ grep "root\|xfs" grep.demo
root         1  0.0  0.6 128164  6824 ?        Ss   Feb26   0:01 /usr/lib/systemd/
systemd --switched-root --system --deserialize 21
root         2  0.0  0.0      0     0 ?        S    Feb26   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Feb26   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Feb26   0:00 [kworker/0:0H]
root       389  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs_mru_cache]
root       390  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-buf/dm-0]
root       391  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-data/dm-0]
root       392  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-conv/dm-0]
root       393  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-cil/dm-0]
root       394  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-reclaim/dm-]
root       395  0.0  0.0      0     0 ?        S<   Feb26   0:00 [xfs-log/dm-0]
```

grep examples:

- If we want to find the lines contain 'ss'
- grep "ss" grep.demo



```
[hao@localhost ~]$ grep ss grep.demo
dbus        627  0.0  0.1  32776  1856 ?        Ssl  Feb26   0:00 /bin/dbus-daemon --
system --address=systemd: --nofork --nopidfile --systemd-activation
root       1009  0.0  0.4 105996  4072 ?        Ss   Feb26   0:00 /usr/sbin/sshd -D
root       2156  0.0  0.5 145700  5276 ?        Ss   Feb26   0:00 sshd: hao [priv]
hao        2160  0.0  0.2 145700  2400 ?        S    Feb26   0:00 sshd: hao@pts/0
```

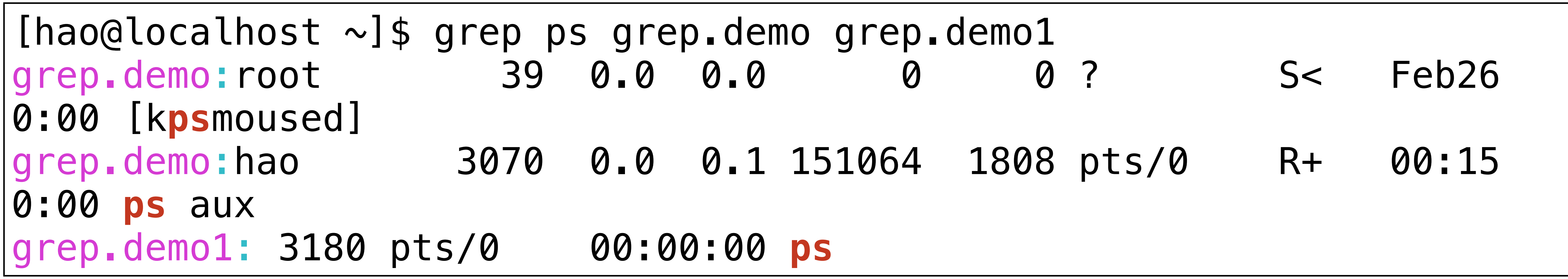
grep examples:

- If we want to find the lines contain 'ss' or 'SS' or 'Ss' or 'sS'
- grep -i "ss" grep.demo

```
[hao@localhost ~]$ grep -i ss grep.demo
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.6 128164  6824 ?        Ss   Feb26   0:01 /usr/lib/systemd/
systemd --switched-root --system --deserialize 21
root       464  0.0  0.3  34988  3788 ?        Ss   Feb26   0:00 /usr/lib/systemd/
systemd-journald
root       483  0.0  0.4 129552  4136 ?        Ss   Feb26   0:00 /usr/sbin/lvmetad -f
root       495  0.0  0.4  46772  4824 ?        Ss   Feb26   0:00 /usr/lib/systemd/
systemd-udevd
root       624  0.0  0.1  24204  1708 ?        Ss   Feb26   0:00 /usr/lib/systemd/
systemd-logind
polkitd    626  0.0  1.2 534892 12884 ?        Ssl  Feb26   0:00 /usr/lib/polkit-1/
polkitd --no-debug
```

grep examples:

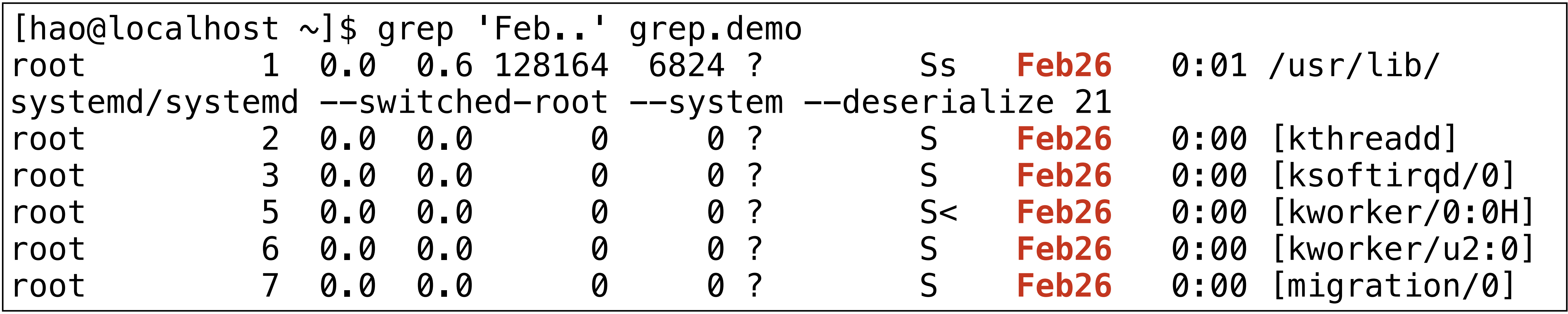
- If we want to find the lines contain 'ps' from multiple files:
- We make another file called grep.demo1 by using command:
- ps > grep.demo1
- grep ps grep.demo grep.demo1



```
[hao@localhost ~]$ grep ps grep.demo grep.demo1
grep.demo:root          39  0.0  0.0      0     0 ?        S<   Feb26
0:00 [kpsmoused]
grep.demo:hao         3070  0.0  0.1 151064  1808 pts/0    R+   00:15
0:00 ps aux
grep.demo1: 3180 pts/0    00:00:00 ps
```

Using grep and regular expression

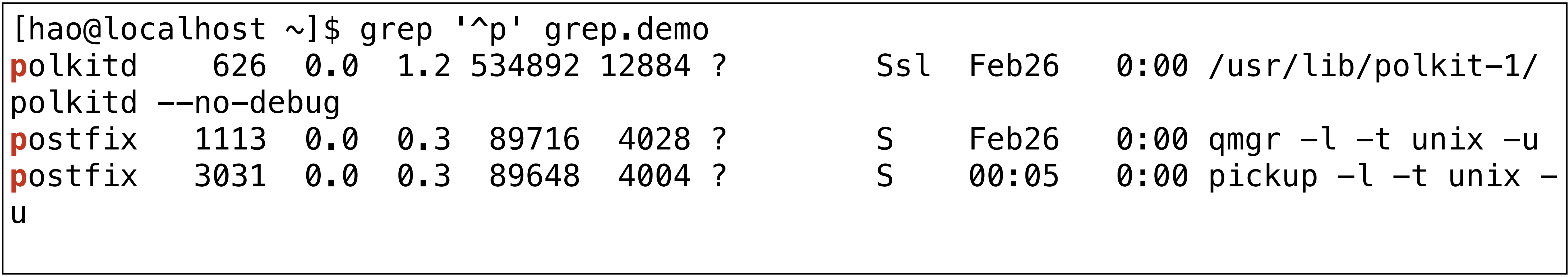
- grep command is usually been used with regular expression.
- If we want to list all the processes started in Feb:
- grep 'Feb..' grep.demo



```
[hao@localhost ~]$ grep 'Feb..' grep.demo
root         1  0.0  0.6 128164  6824 ?        Ss   Feb26   0:01 /usr/lib/
systemd/systemd --switched-root --system --deserialize 21
root         2  0.0  0.0      0     0 ?        S    Feb26   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Feb26   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Feb26   0:00 [kworker/0:0H]
root         6  0.0  0.0      0     0 ?        S    Feb26   0:00 [kworker/u2:0]
root         7  0.0  0.0      0     0 ?        S    Feb26   0:00 [migration/0]
```

Using grep and regular expression

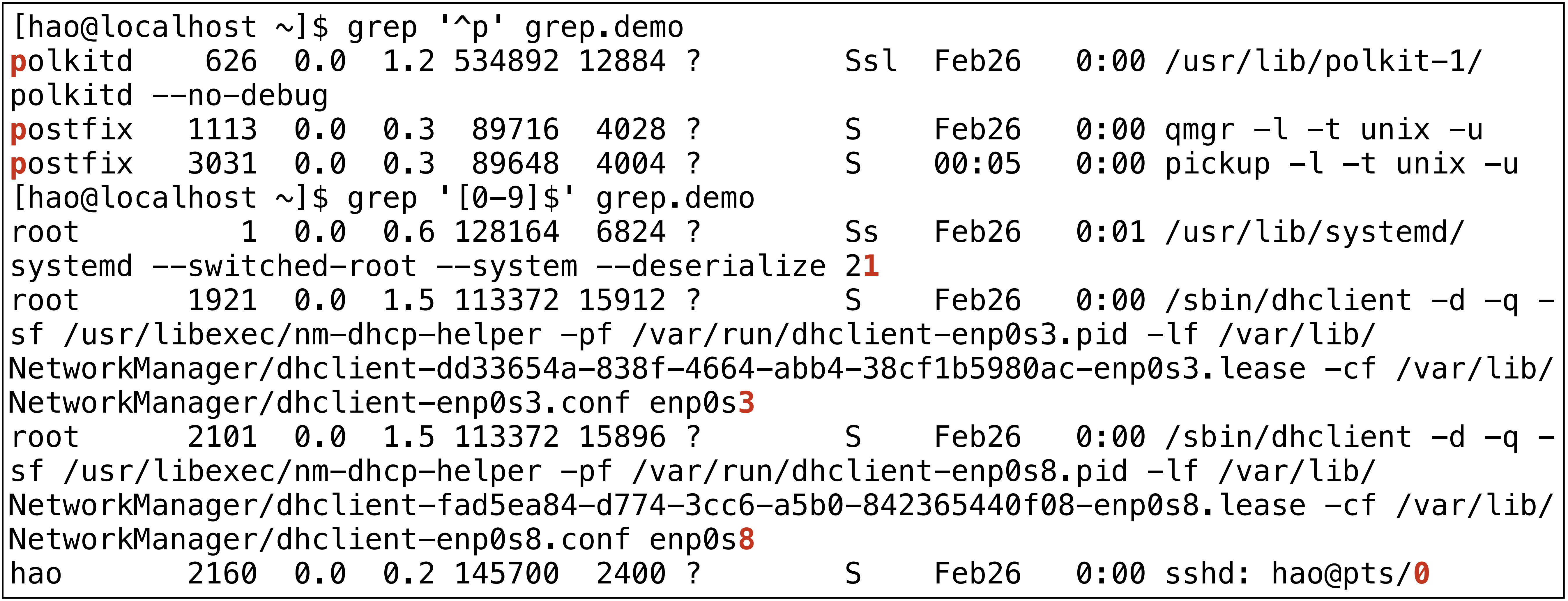
- grep command is usually been used with regular expression.
- If we want to list all the lines start with p :
- grep '^p' grep.demo



```
[hao@localhost ~]$ grep '^p' grep.demo
polkitd    626  0.0  1.2 534892 12884 ?        Ssl  Feb26   0:00 /usr/lib/polkit-1/
polkitd --no-debug
postfix   1113  0.0  0.3  89716  4028 ?        S    Feb26   0:00 qmgr -l -t unix -u
postfix   3031  0.0  0.3  89648  4004 ?        S    00:05   0:00 pickup -l -t unix -
u
```

Using grep and regular expression

- grep command is usually been used with regular expression.
- If we want to list all the lines end with a number :
- grep '[0-9]$' grep.demo



```
[hao@localhost ~]$ grep '^p' grep.demo
polkitd    626  0.0  1.2 534892 12884 ?        Ssl  Feb26   0:00 /usr/lib/polkit-1/
polkitd --no-debug
postfix   1113  0.0  0.3  89716  4028 ?        S    Feb26   0:00 qmgr -l -t unix -u
postfix   3031  0.0  0.3  89648  4004 ?        S    00:05   0:00 pickup -l -t unix -u
[hao@localhost ~]$ grep '[0-9]$' grep.demo
root         1  0.0  0.6 128164  6824 ?        Ss   Feb26   0:01 /usr/lib/systemd/
systemd --switched-root --system --deserialize 21
root      1921  0.0  1.5 113372 15912 ?        S    Feb26   0:00 /sbin/dhclient -d -q -
sf /usr/libexec/nm-dhcp-helper -pf /var/run/dhclient-enp0s3.pid -lf /var/lib/
NetworkManager/dhclient-dd33654a-838f-4664-abb4-38cf1b5980ac-enp0s3.lease -cf /var/lib/
NetworkManager/dhclient-enp0s3.conf enp0s3
root      2101  0.0  1.5 113372 15896 ?        S    Feb26   0:00 /sbin/dhclient -d -q -
sf /usr/libexec/nm-dhcp-helper -pf /var/run/dhclient-enp0s8.pid -lf /var/lib/
NetworkManager/dhclient-fad5ea84-d774-3cc6-a5b0-842365440f08-enp0s8.lease -cf /var/lib/
NetworkManager/dhclient-enp0s8.conf enp0s8
hao       2160  0.0  0.2 145700  2400 ?        S    Feb26   0:00 sshd: hao@pts/0
```

Exercise

- Please create a file contain the following information:

ID	Name	City	Advisor	DOB	Year	GPA
282102	Andy	New Haven	Lisa	01/23/99	Sr	3.5
281734	Zoe	Bridgeport	Hao	05/14/00	Jr	3.2
292342	Alex	Cheshire	hao	12/25/01	Fresh	2.0
284323	Eric	Hartford	lisa	02/26/00	Soph	2.5
274342	Bill	Orange	Imad	04/12/99	Sr	4.0
293232	Alice	New Haven	IMAD	07/04/98	Jr	1.8
213852	Lee	bridgeport	lisa	06/22/02	Fresh	3.8

Questions:

- Find the students whose advisor is Lisa
- Find the students come from orange
- Find the students come from New Haven
- Find the students do not come from New Haven
- Find the students born after 2000
- Find the students have GPA higher than 3
- Find the students whose advisor is Hao and have GPA higher than 3
- Find the students with their ID start with 28

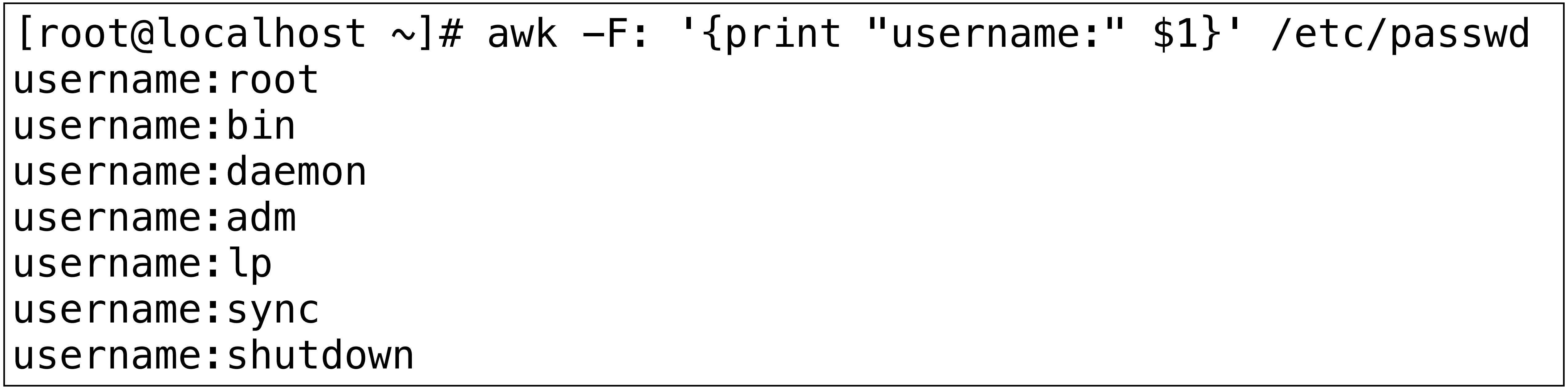
awk: text processing and data extraction tool

- awk: is a language for processing text files
- Syntax:

-
  ```
  awk [options] ‘command’ input_file
  ```
- Commonly used options:
  - F: field separator
- commands:
  - build-in functions are enclosed in {}

awk: text processing and data extraction tool

- Example: separate field in /etc/passwd file and print out username:
- `awk -F: ‘{print ”username:” $1}’ /etc/passwd`
- $1: the first field in the line
- $0: the whole line



```
[root@localhost ~]# awk -F: '{print "username:" $1}' /etc/passwd
username:root
username:bin
username:daemon
username:adm
username:lp
username:sync
username:shutdown
```

awk with regular expression

- Example: separate field in /etc/passwd file and print out root :
- `awk -F: ‘/root/{print ”username:” $1}’ /etc/passwd`

```
[root@localhost ~]# awk -F: '/root/{print "username:" $1}' /etc/
passwd
username:root
username:operator
```

awk with regular expression

- Example: separate field in /etc/passwd file and print out root :
- we only want first line
- `awk -F: ‘$1==“root”{print ”username:” $1}’ /etc/passwd`

```
[root@localhost ~]# awk -F: '$1=="root"{print "username:" $1}' /
etc/passwd
username:root
```

Exercise:

- Print the students name whose advisor is Lisa
- Print the name and DOB of students who come from orange
- Print the name and GPA of students who come from New Haven
- Print the name and hometown of students do not come from New Haven
- Print the DOB of students born after 2000
- Print the name, GPA, and advisor of students have GPA higher than 3
- Print the GPA and name of students whose advisor is Hao and have GPA higher than 3
- Print the name and year of students with GPA less than 3

Solution for Exercise 1:

```
grep -i lisa students
grep -i orange students
grep "New Haven" students
grep -v "New Haven" students
grep "[0-9][0-9]/[0-9][0-9]/0[0-9]" students
grep "[3-9]\.[0-9]" students
grep "[3-9]\.[0-9]" students|grep -i hao
grep "^28" students
```

Solution for Exercise 2:

```
awk '/[Ll]isa/{print $2}' students
```

I can also print out the header before names:

```
head -1 students|awk '{print $2}';awk '/[lL]isa/{print
$2}' students

awk '/Orange/{print $2" " $5}' students
awk '/New Haven/{print $2" " $8}' students
grep -v "New Haven" students|awk '{print $2" “$3}'
awk '/[0-9][0-9]\/[0-9][0-9]\/0[0-9]/{print $5}'
students
awk '/[34]\.[0-9]$/{if ($4=="Haven") print $2" "$5" "$8;
else print $2" "$4" "$7}' students
awk '/[34]\.[0-9]$/{if ($4=="Hao"||$4=="hao") print $7"
"$2;}' students
awk '/[0-2]\.[0-9]$/{if($4=="Haven") print $2" "$6;else
print $2" "$5}' students
```