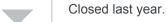
Print a file, skipping the first X lines, in Bash [duplicate]

Asked 11 years, 10 months ago Active 2 days ago Viewed 439k times



This question already has answers here:

How can I remove the first line of a text file using bash/sed script? (16 answers)





I have a very long file which I want to print, skipping the first 1,000,000 lines, for example.

I looked into the cat man page, but I did not see any option to do this. I am looking for a command to do this or a simple Bash program.

linux bash printing skip





13 Answers





You'll need tail. Some examples:

```
951
```

```
$ tail great-big-file.log
< Last 10 lines of great-big-file.log >
```



If you really need to SKIP a particular number of "first" lines, use



```
$ tail -n +<N+1> <filename>
< filename, excluding first N lines. >
```

That is, if you want to skip N lines, you start printing line N+1. Example:

```
$ tail -n +11 /tmp/myfile
< /tmp/myfile, starting at line 11, or skipping the first 10 lines. >
```

If you want to just see the last so many lines, omit the "+":

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



edited Dec 14 '18 at 23:27



rogerdpack **47.9k** 30 204 322



answered Mar 3 '09 at 2:24 SingleNegationEliminati

> **133k** 25 242 282

- 67 Or "tail --lines=+<LinesToSkip> ..." for the readable-commands crowd :-) paxdiablo Mar 3 '09 at 2:34
- 30 in centos 5.6 tail -n +1 shows the whole file and tail -n +2 skips first line. strange. The same for tail -c +<num> . - NickSoft Sep 1 '11 at 10:23 ▶
- 14 @JoelClark No, @NickSoft is right. On Ubuntu, it's tail -n +<start number>, I just tested it. So tail -n +1 won't skip anything, but start from the first line instead. - Andres F. Aug 22 '12 at 14:36
- 22 I can confirm that tail -n +2 is required to skip the first line on Darwin/Mac OS X as well. morgant Mar 24 '14 at 16:40
- this must be outdated, but, tail -n+2 OR tail -n +2 works, as with all short commands using getopt, you can run the parameter right next to it's switch, providing that the switch is the last in the group, obviously a command like tail -nv+2 would not work, it would have to be tail -vn+2. if you dont believe me try it yourself. - osirisgothra May 3 '14 at 11:35



Easiest way I found to remove the first ten lines of a file:

125

\$ sed 1,10d file.txt



In the general case (where x is the number of initial lines to delete, credit to commenters and editors for this):

\$ sed 1,Xd file.txt

edited 2 days ago

answered Oct 17 '12 at 7:17



David Parks 39

- 13 In the more general case, you'd have to use sed 1,Xd where X is the number of initial lines to delete, with X greater than 1. - Acumenus Dec 24 '13 at 0:10
- This makes more sense if you don't know how long the file is and don't want to tell tail to print the last 100000000 lines. - springloaded Aug 29 '18 at 15:06

@springloaded if you need to know the number of lines in the file, 'wc -l' will easily give it to you -Mike Pennington Jun 10 at 11:21



If you have GNU tail available on your system, you can do the following:

104 tail -n +1000001 huge-file.log

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.





If the first character of K (the number of bytes or lines) is a `+', print beginning with the Kth item from the start of each file.

Thus, as noted in the comment, putting +1000001 starts printing with the first item after the first 1,000,000 lines.





answered Mar 3 '09 at 2:28

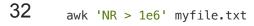


50.5k 21 115 141

Works for BSD tail too (OS X) – Lloeki Nov 17 '16 at 13:59



A less verbose version with AWK:





1

But I would recommend using integer numbers.



answered Apr 26 '13 at 14:31



7 useful if you need to skip some lines in the middle of the file, e.g., awk '!(5 < NR && NR < 10)' − arekolek Jul 28 '16 at 12:24 ✓



If you want to skip first two line:

23 tail -n +3 <filename>



1

If you want to skip first x line:

tail -n +\$((x+1)) <filename>

edited Apr 13 at 2:29



answered Jul 9 '13 at 18:10



saipraneeth 347 2 2

2 This is somewhat misleading because someone may interpret (x+1) literally. For example, for x=2, they may type either (2+1) or even (3) neither of which would work. A better way to write it might be: To

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.





Use the <u>sed_delete_command</u> with a <u>range address</u>. For example:

18

sed 1,100d file.txt # Print file.txt omitting lines 1-100.



Alternatively, if you want to only print a known range, use the print command with the -n flag:

sed -n 201,300p file.txt # Print lines 201-300 from file.txt

This solution should work reliably on all Unix systems, regardless of the presence of GNU utilities.

edited Apr 13 at 2:31



Peter Mortensen **27.2k** 21 93

answered Dec 2 '16 at 16:19



maerics

129k 36 237

Most readily usable answer for both cli and scripting. - cerd Dec 1 '17 at 1:37



If you want to see the first 10 lines you can use sed as below:

15

sed -n '1,10 p' myFile.txt



Or if you want to see lines from 20 to 30 you can use:

1

sed -n '20,30 p' myFile.txt

edited Apr 13 at 2:28



Peter Mortensen **27.2k** 21 93

answered Dec 20 '12 at 10:04





Just to propose a sed alternative.:) To skip first one million lines, try | sed '1,1000000d'.

Example: 14



\$ perl -wle 'print for (1..1 000 005)'|sed '1,1000000d' 1000001



1000002



1000004 1000005

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



2 @Marlon, sorry but that's wrong. That only works for 1d. If, for example, you use it on 2d, you'll delete only line 2. It doesn't delete the range of lines. − Acumenus Dec 24 '13 at 17:19 ✓

@A-B-B sorry, meant to say that this was the easiest solution by far which is why I +1 it not trying to correct the author. – Marlon Jan 14 '14 at 19:40



Use:

13

sed -n '1d;p'



This command will delete the first line and print the rest.



edited Apr 13 at 2:32



Peter Mortensen **27.2k** 21 93 123



Soroush Pouryazdian

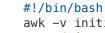
answered Aug 3 '18 at 16:23

better than tail imo, since we don't have to know the number of lines to be tail-ed. we just remove the 1st line and that's all – Tom Jan 31 at 15:24



This shell script works fine for me:

10



}' \$3

awk -v initial_line=\$1 -v end_line=\$2 '{
 if (NR >= initial_line && NR <= end_line)
 print \$0</pre>



Used with this sample file (file.txt):

one two three four

five six

The command (it will extract from second to fourth line in the file):

```
edu@debian5:~$./script.sh 2 4 file.txt
```

Output of this command:

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



Of course, you can improve it, for example by testing that all argument values are the expected :-)

answered Mar 31 '09 at 13:28



sourcerebels

++ for using awk, which is oh so marginally more portable than tail – guns Mar 31 '09 at 13:42



You can do this using the head and tail commands:



head -n <num> | tail -n <lines to print>



4

where num is 1e6 + the number of lines you want to print.



answered Mar 3 '09 at 2:25



Dana the Sane

13.5k 8 51 76

Not the most efficient answer since you'd need to do a "wc -I" on the file to get a line count, followed by an addition to add the million :-). You can do it with just "tail". - paxdiablo Mar 3 '09 at 2:43

I'm not sure, my understanding was that 1e6 would be known at the time of calling. Counting backwards isn't the fastest though. - Dana the Sane Mar 3 '09 at 3:11



cat < File > | awk '{if(NR > 6) print \$0}'







edited Nov 21 '12 at 20:34



C. A. McCann

answered Nov 21 '12 at 20:14



aamadeo

This is a syntax error in bash — in what shell does it work? – G-Man Says 'Reinstate Monica' May 18 '17 at 4:51

I run this in bash. The < and > are not part of the command, the name of the file should replace "< File >" aamadeo May 19 '17 at 13:37

awk 'NR > 6 {print}' is sufficient... no need for the if or the \$0. - CSTobey Jan 9 '19 at 20:45



I needed to do the same and found this thread.

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.





The more +lines worked nicely on the prompt, but it turned out it behaved totally different when run in headless mode (cronjob).

I finally wrote this myself:

```
skip=5
FILE="/tmp/filetoprint"
tail -n$((`cat "${FILE}" | wc -l` - skip)) "${FILE}"
```

answered May 28 '09 at 17:30 frater

- Correct link of <u>Useless Use of Cat Award</u>. The previous is replaced by advert. kub1x Jul 26 '17 at 13:01
- @kub1x I don't think "cat" here is useless, as "cat | wc -l" produces different output than simple "wc -l". The former is suitable for arithmetic operations, the latter is not. - Jack Jan 15 '18 at 10:14
 - @Jack I wasn't judging the use of cat, but only fixing a link in a comment, that led to a dead page. The original comment must have been deleted. Anyways, thanks for pointing that out. - kub1x Jan 15 '18 at 11:55
- @kub1x You know? After reading the link now I think that the use of "cat" here is wrong:) It should has been something like "wc -l < \${FILE}", saving some overhead time/memory (new process creation, pipelining I/O,...). Thanks, I've learned something new – Jack Jan 16 '18 at 9:43 ▶



Mighly active question. Earn 10 reputation in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.

