Yandex

Unix Command Line

File content exploration
Part II

How can we modify text files?

Files editing commands and tools

- vim
- cut
- tr
- sort
- awk
- special symbolic: |, ||, &, &&, >, >>

vim

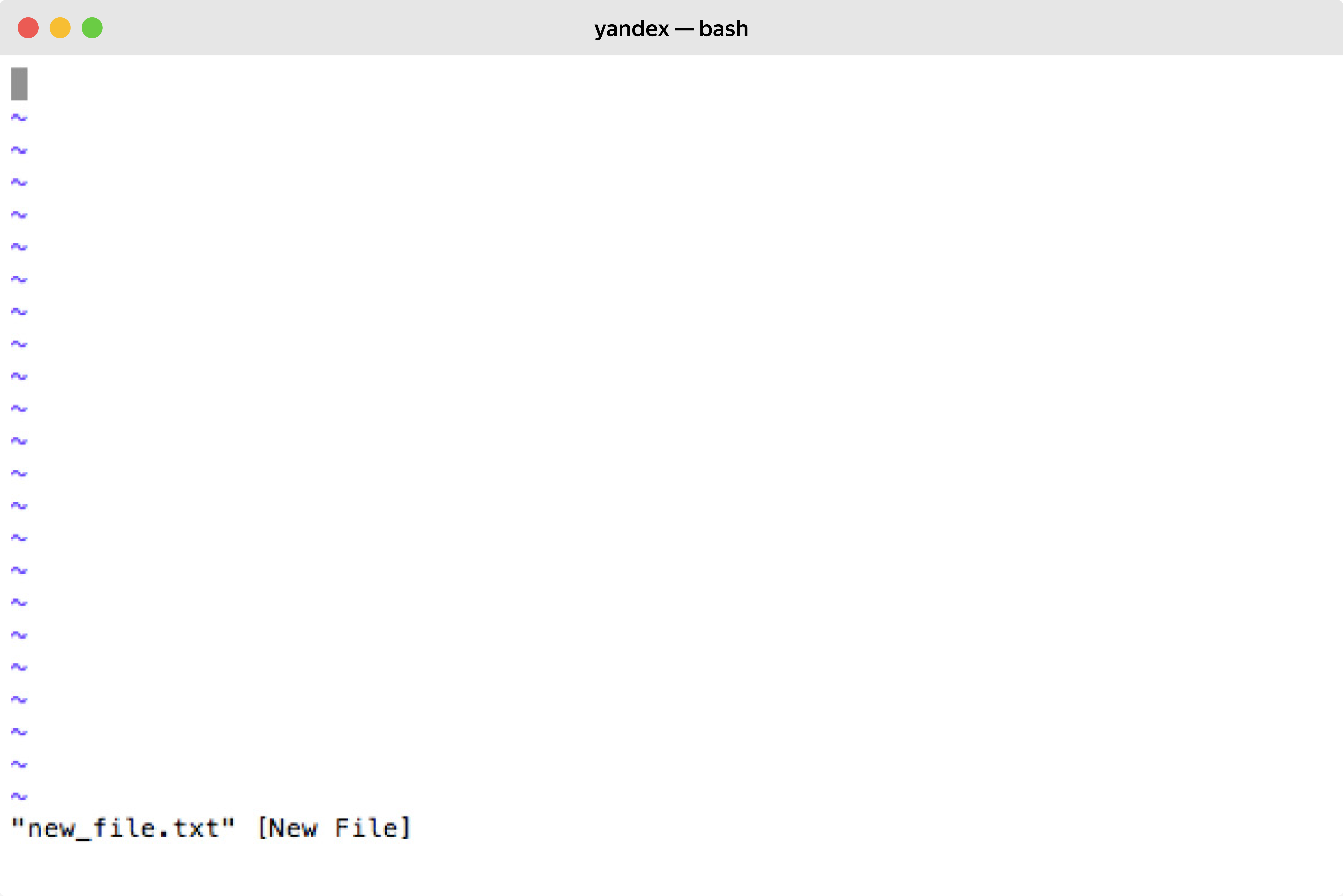
- an advanced text editor
- improved version of vi redactor
- efficient text (including programming code) editing

```
vim [options] [filelist]
```

vim

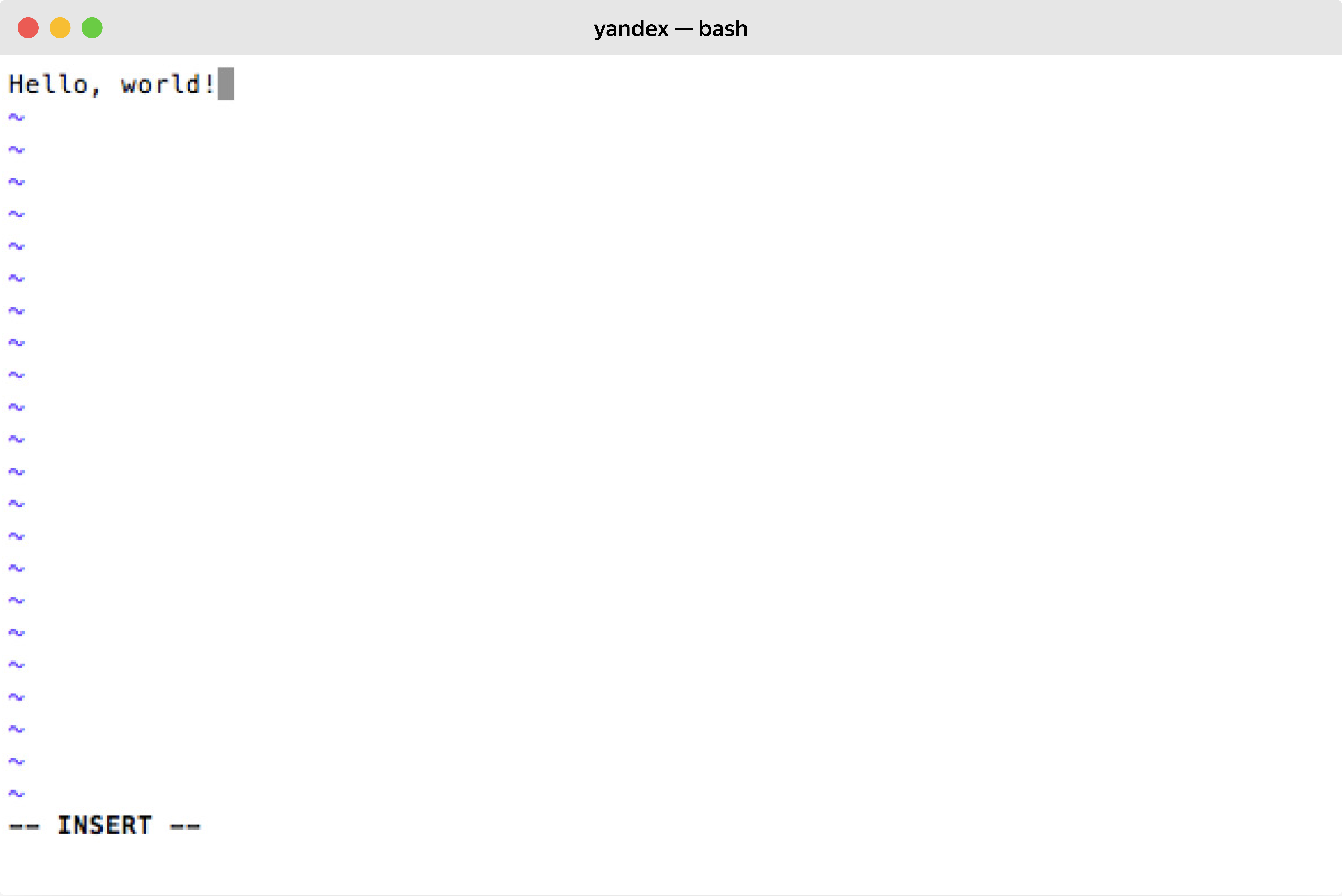
- to start vim: ”vim filename”
- to edit file: “:i” (insert)
- to save file and quit vim: “:wq”
- to quit vim without file saving: “:!q”

vim



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"new_file.txt" [New File]
```

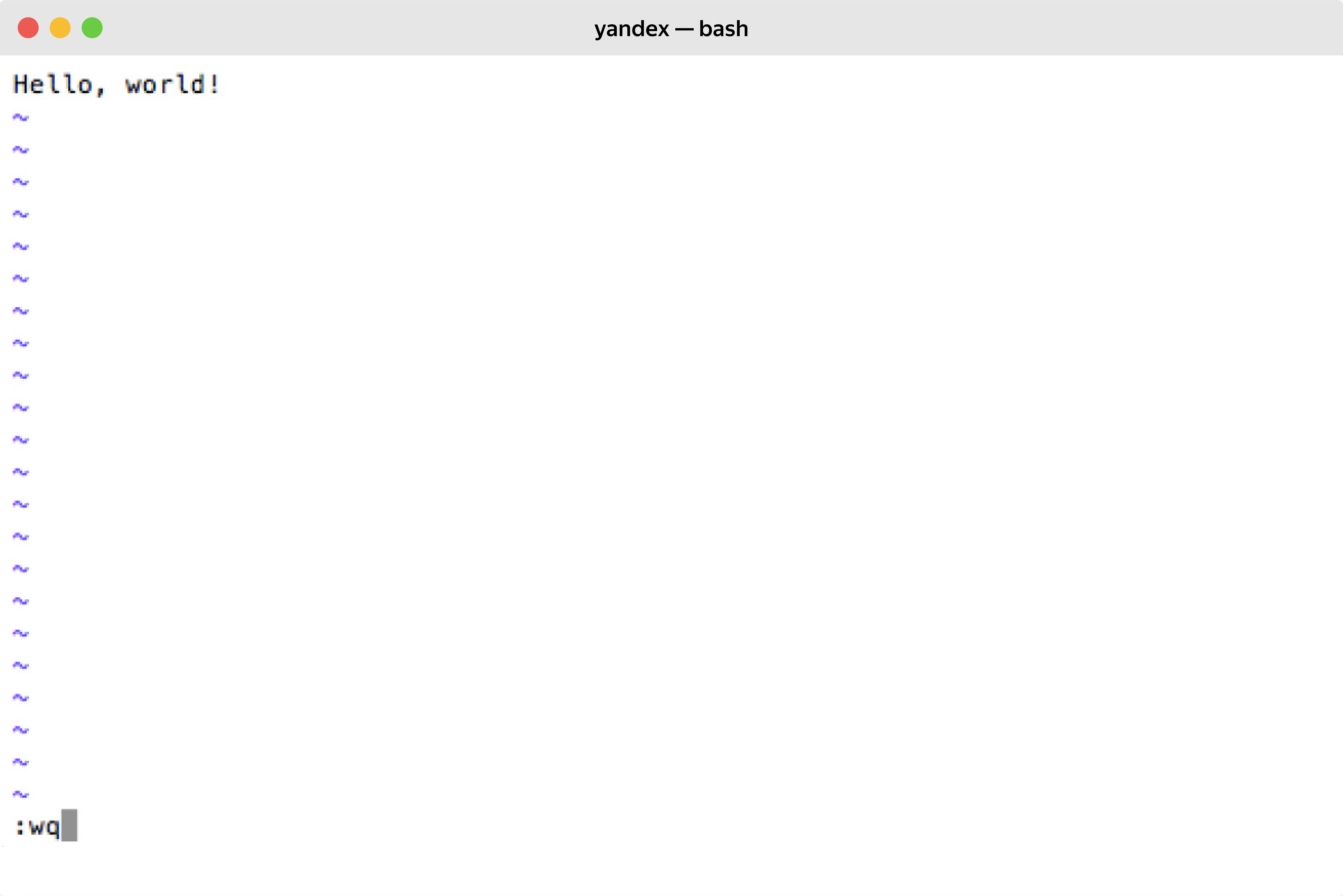
vim



yandex — bash

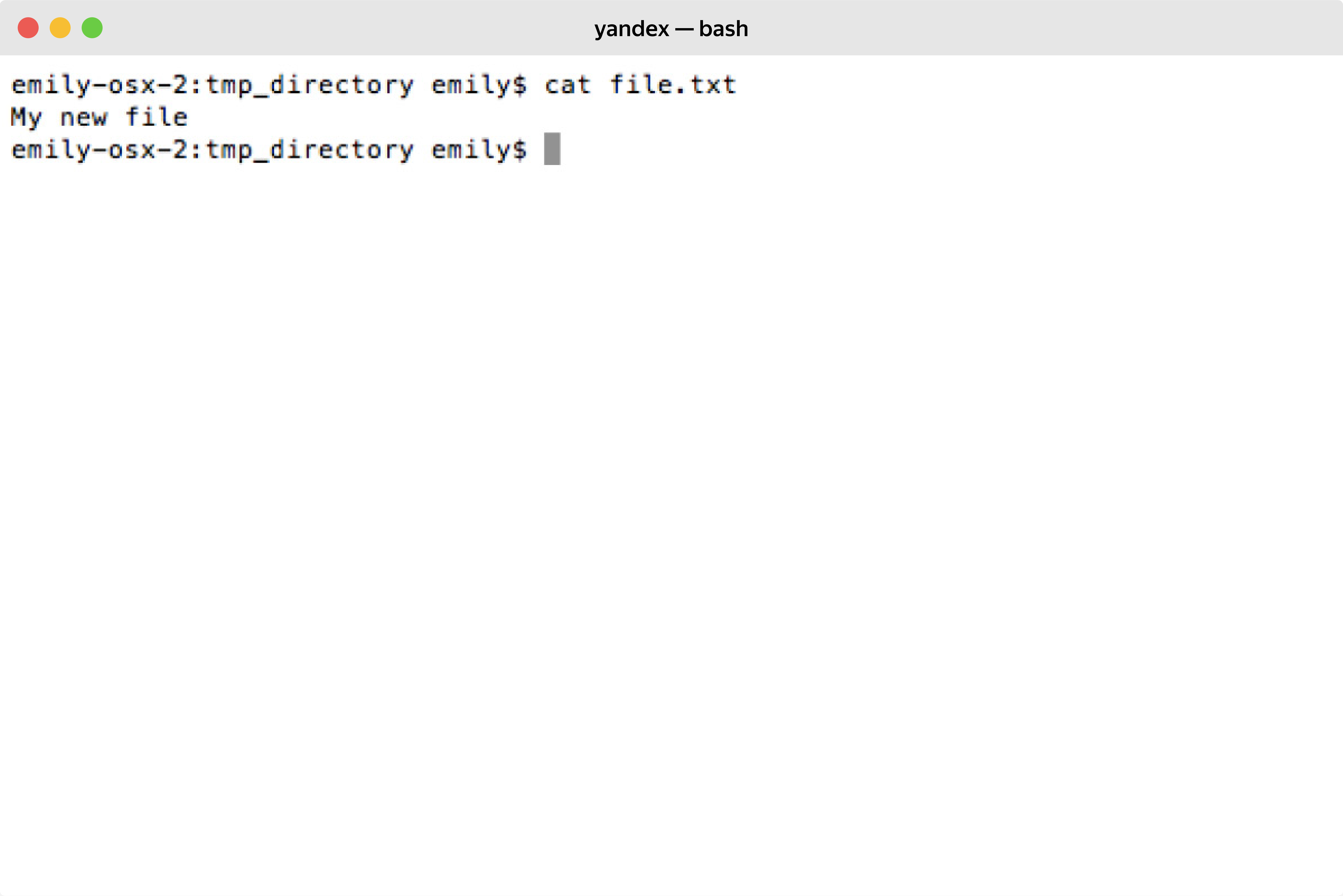
```
Hello, world!
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

vim



```
Hello, world!
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

vim



```
emily-osx-2:tmp_directory emily$ cat file.txt
My new file
emily-osx-2:tmp_directory emily$ 
```

vim

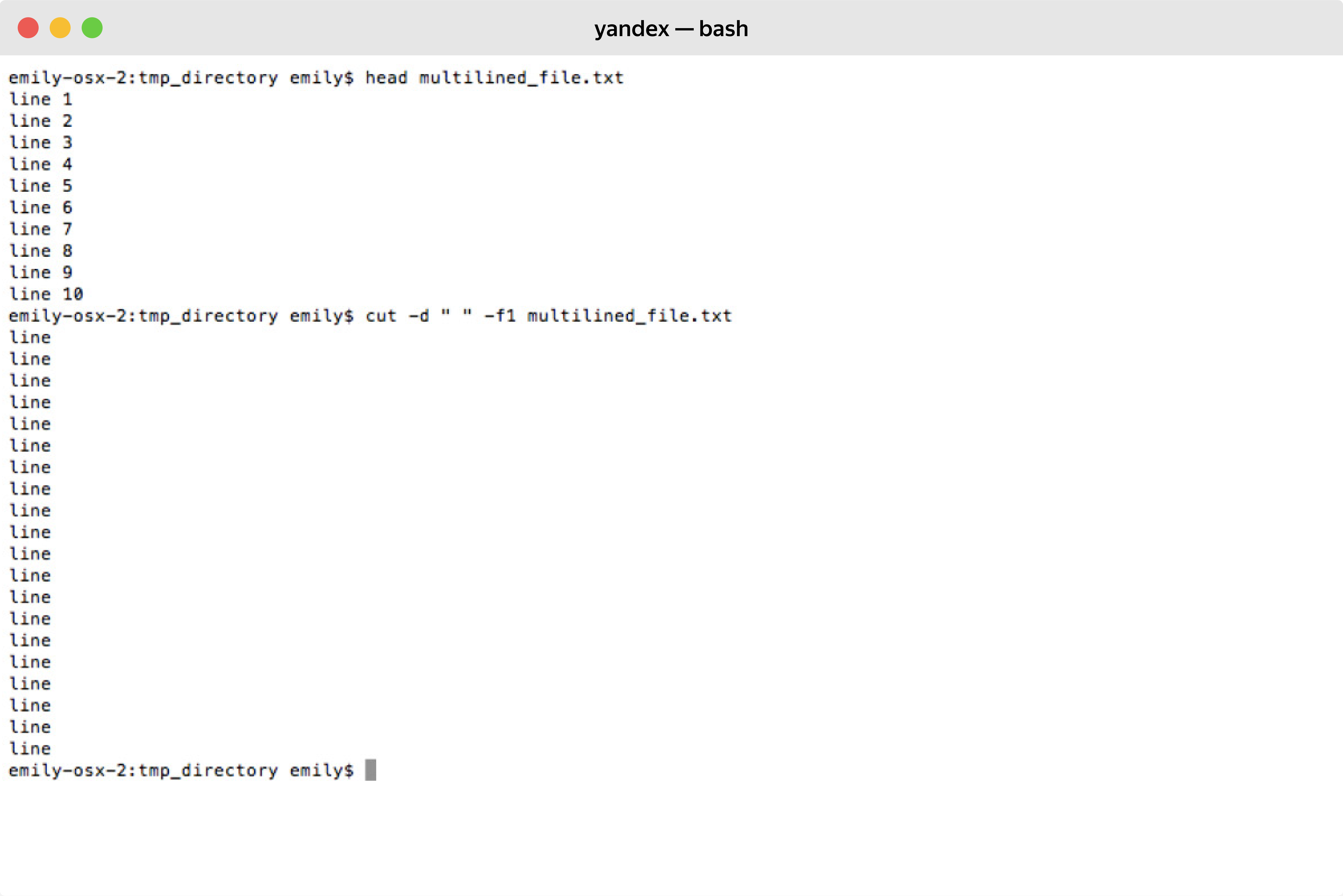
- .vimrc file with vim configuration
- example:
  - colorscheme darkblue
  - set tabstop=4
  - set shiftwidth=4
  - set smarttab
  - set expandtab

cut

- drops sections of each line of a file/files

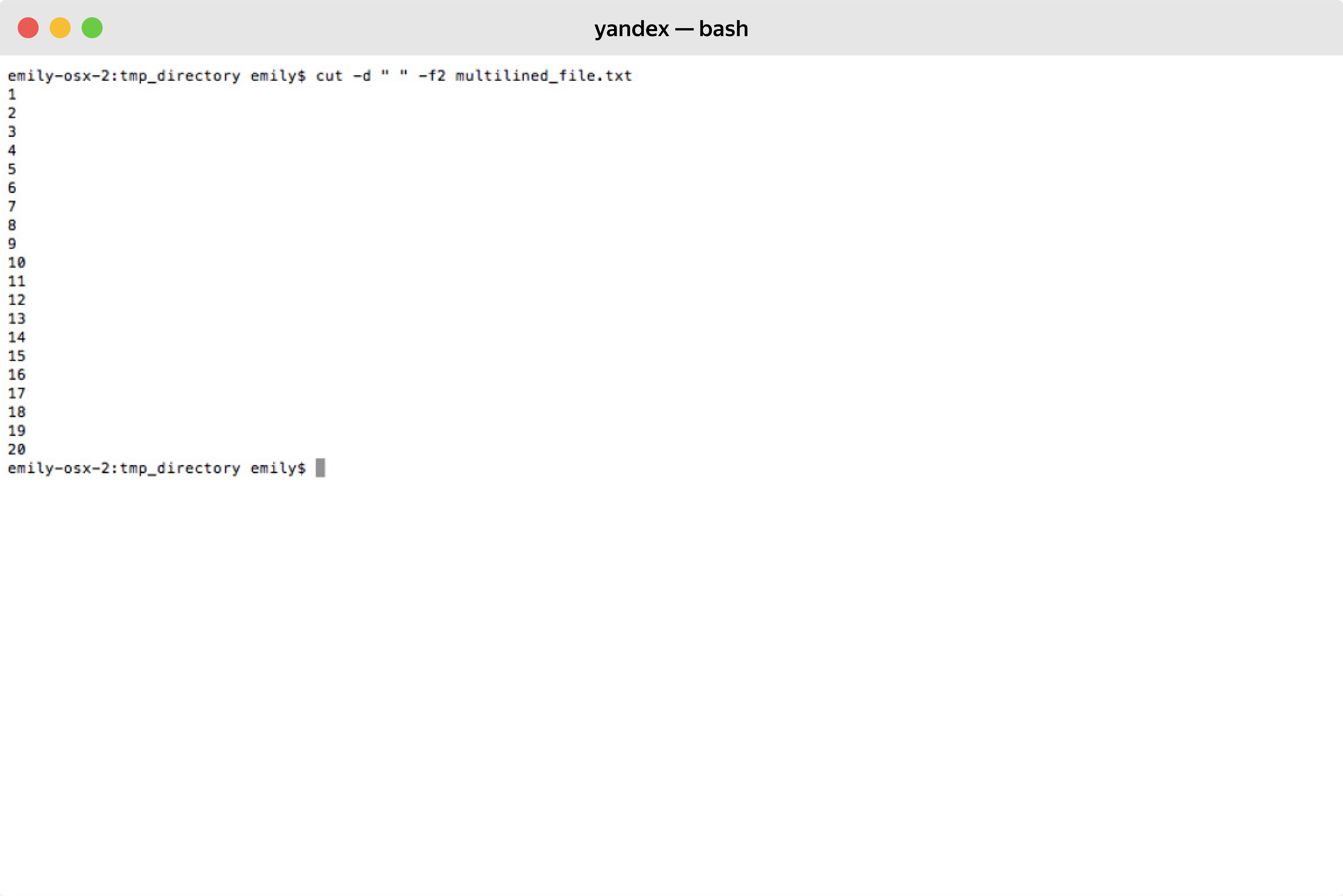
```
cut OPTION... [FILE]...
```

cut -d -f



```
emily-osx-2:tmp_directory emily$ head multilined_file.txt
line 1
line 2
line 3
line 4
line 5
line 6
line 7
line 8
line 9
line 10
emily-osx-2:tmp_directory emily$ cut -d " " -f1 multilined_file.txt
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
emily-osx-2:tmp_directory emily$ 
```

cut -d -f



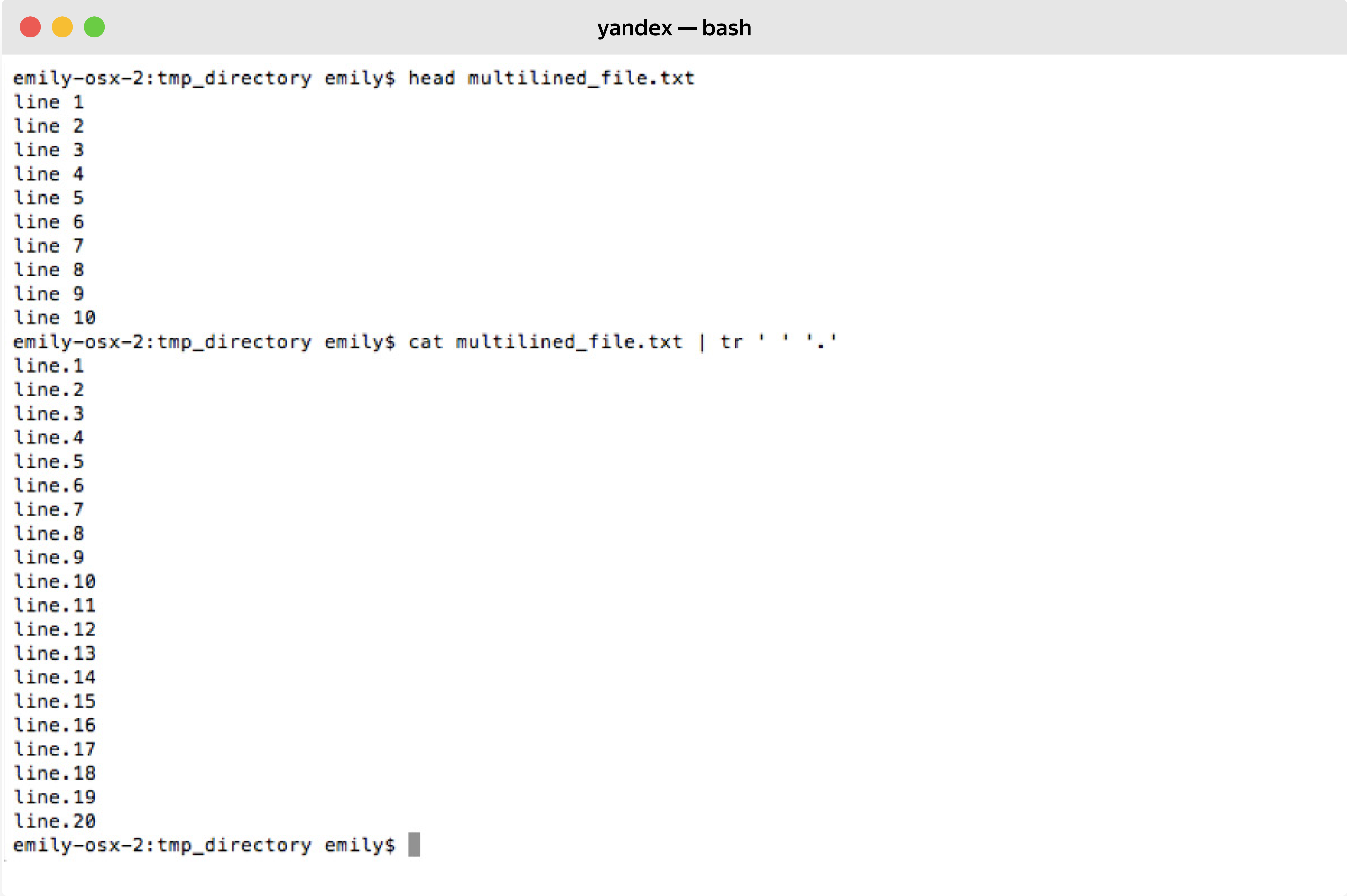
```
emily-osx-2:tmp_directory emily$ cut -d " " -f2 multilined_file.txt
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
emily-osx-2:tmp_directory emily$ 
```

tr

- translates one set of characters to another

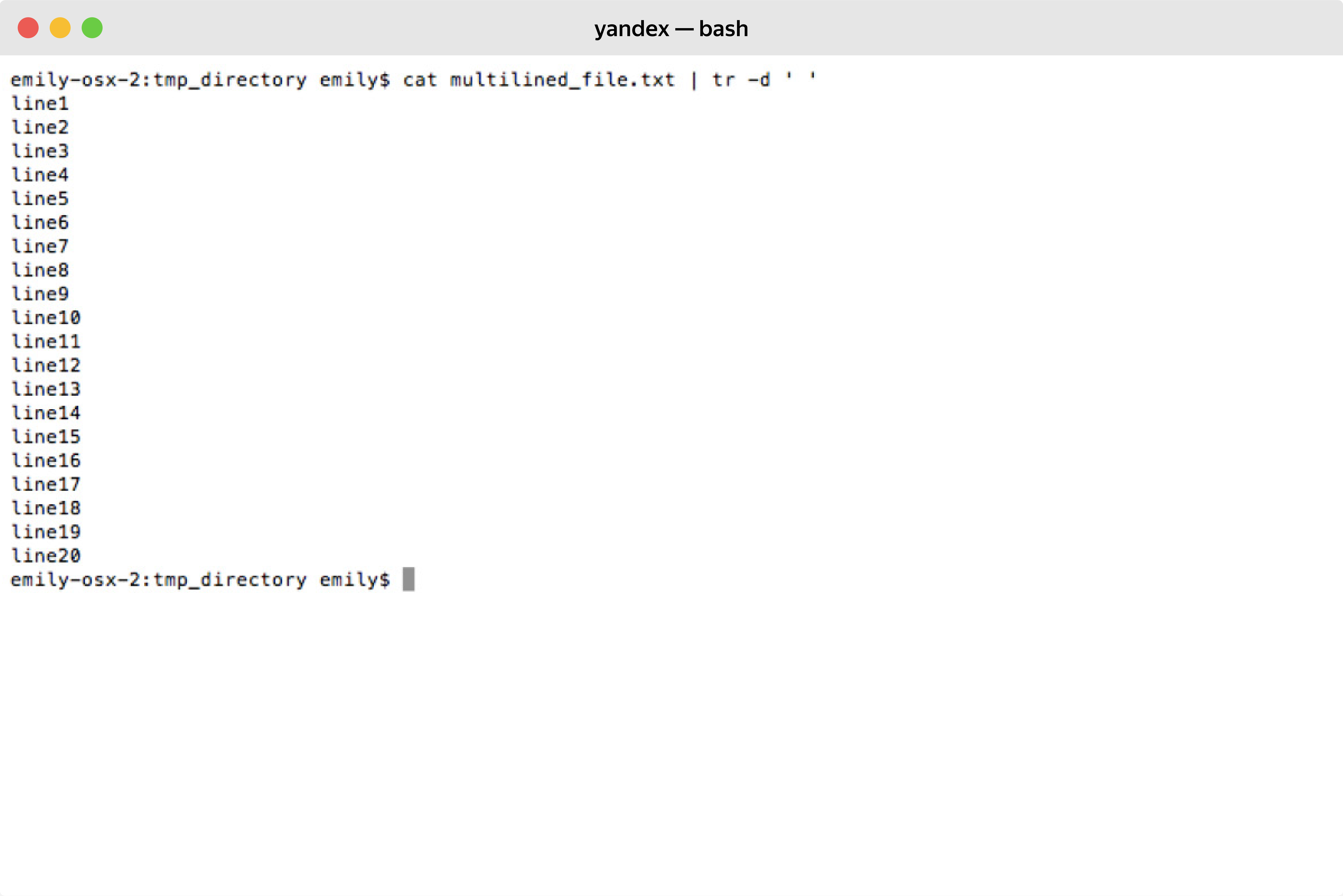
```
tr [-Ccsu] string1 string2
```

tr



```
emily-osx-2:tmp_directory emily$ head multilined_file.txt
line 1
line 2
line 3
line 4
line 5
line 6
line 7
line 8
line 9
line 10
emily-osx-2:tmp_directory emily$ cat multilined_file.txt | tr ' ' '.'
line.1
line.2
line.3
line.4
line.5
line.6
line.7
line.8
line.9
line.10
line.11
line.12
line.13
line.14
line.15
line.16
line.17
line.18
line.19
line.20
emily-osx-2:tmp_directory emily$ 
```

tr -d



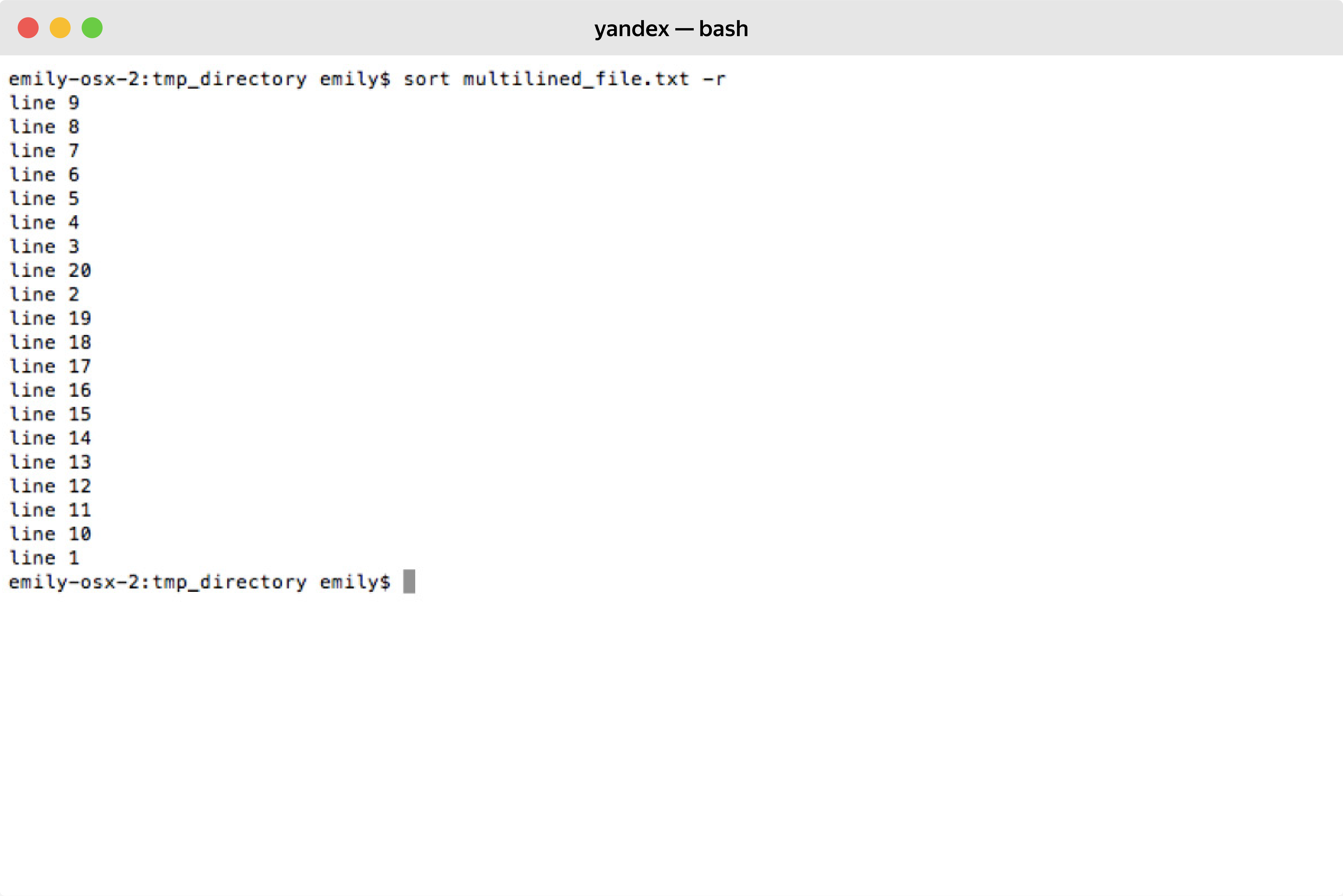
```
emily-osx-2:tmp_directory emily$ cat multilined_file.txt | tr -d ' '
line1
line2
line3
line4
line5
line6
line7
line8
line9
line10
line11
line12
line13
line14
line15
line16
line17
line18
line19
line20
emily-osx-2:tmp_directory emily$ 
```

sort

- sorts the contents of a text file

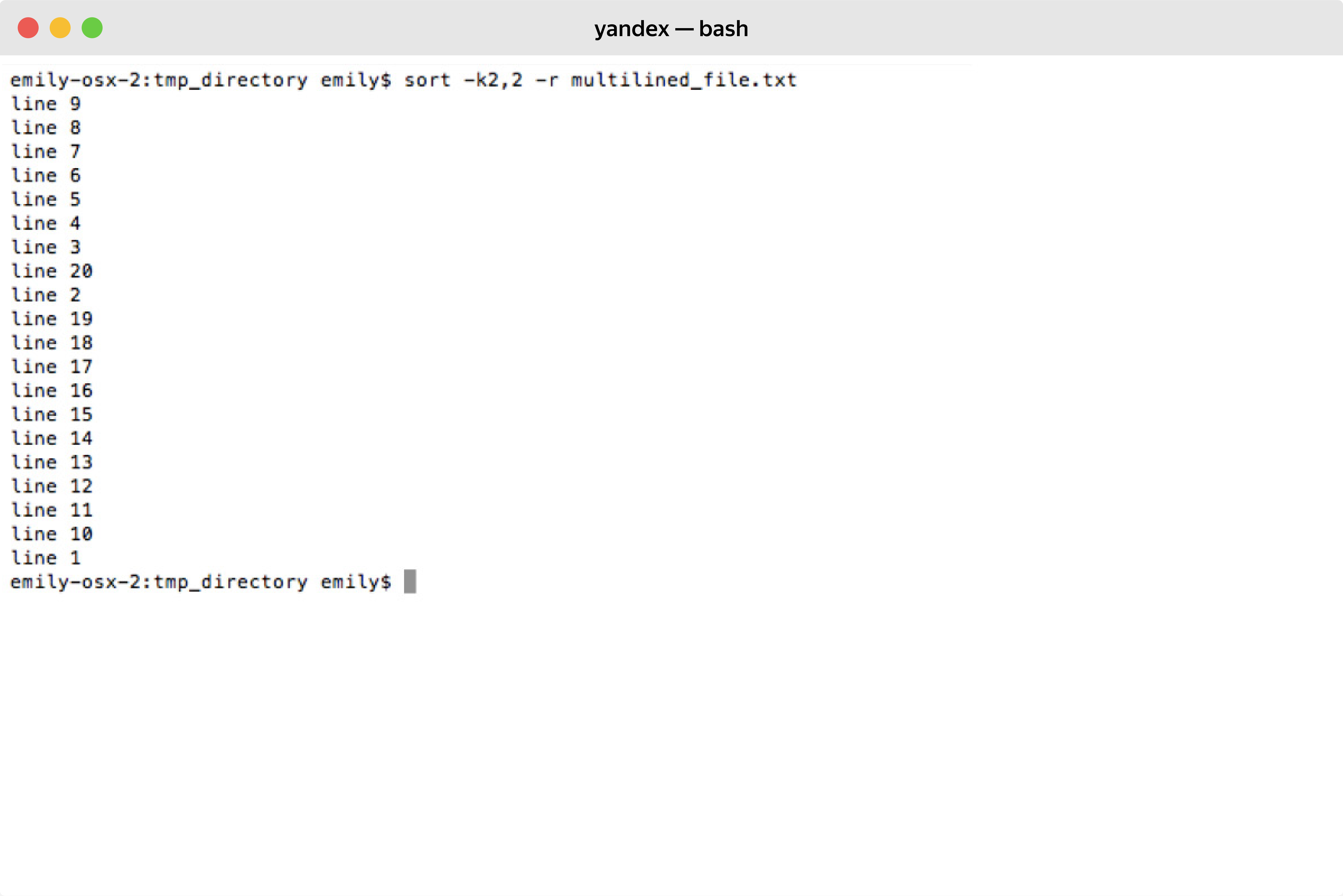
  sort [OPTION]... [FILE]...

sort -r



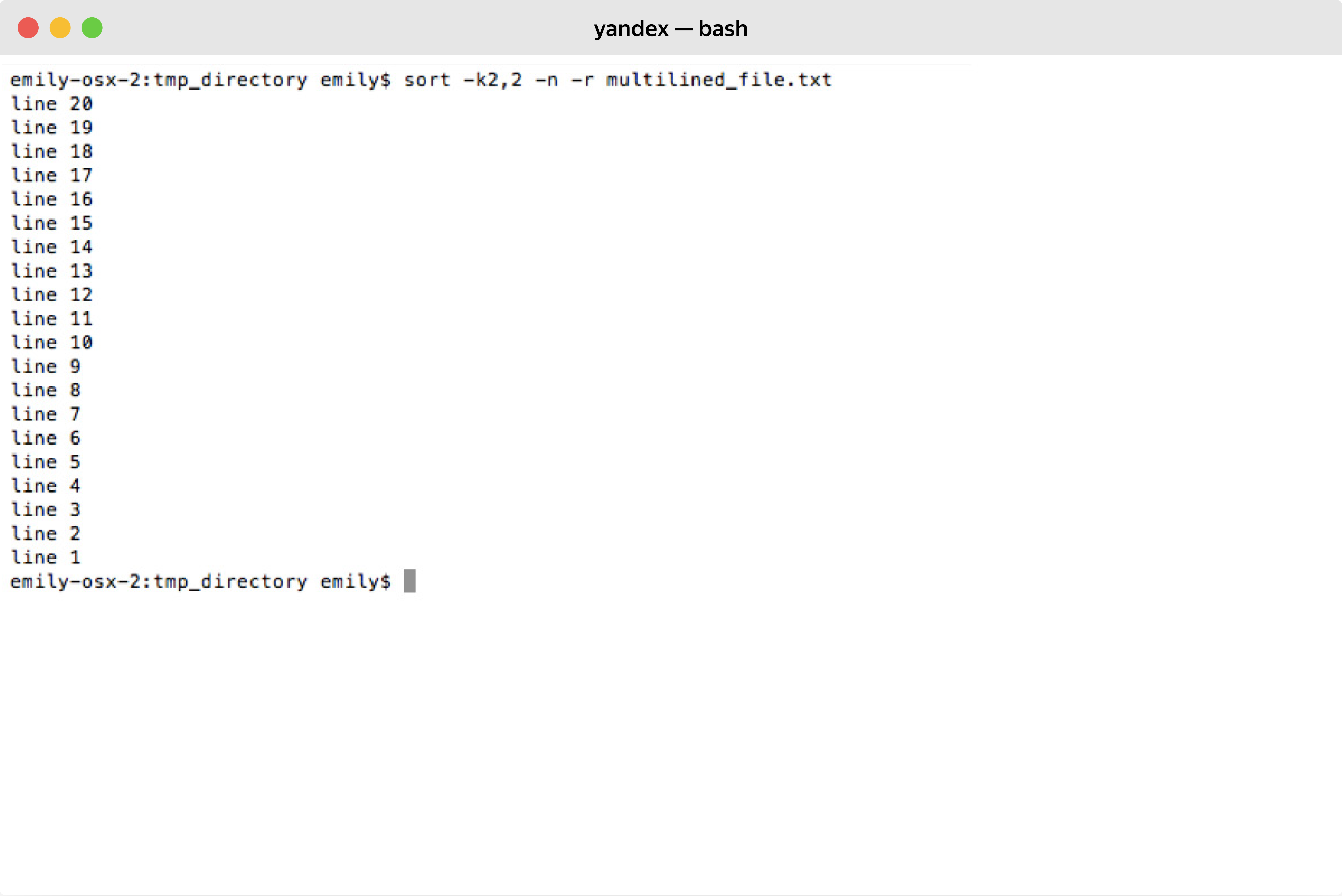
```
emily-osx-2:tmp_directory emily$ sort multilined_file.txt -r
line 9
line 8
line 7
line 6
line 5
line 4
line 3
line 20
line 2
line 19
line 18
line 17
line 16
line 15
line 14
line 13
line 12
line 11
line 10
line 1
emily-osx-2:tmp_directory emily$ 
```

sort -r -k



```
emily-osx-2:tmp_directory emily$ sort -k2,2 -r multilined_file.txt
line 9
line 8
line 7
line 6
line 5
line 4
line 3
line 20
line 2
line 19
line 18
line 17
line 16
line 15
line 14
line 13
line 12
line 11
line 10
line 1
emily-osx-2:tmp_directory emily$ 
```

sort -r -k -n



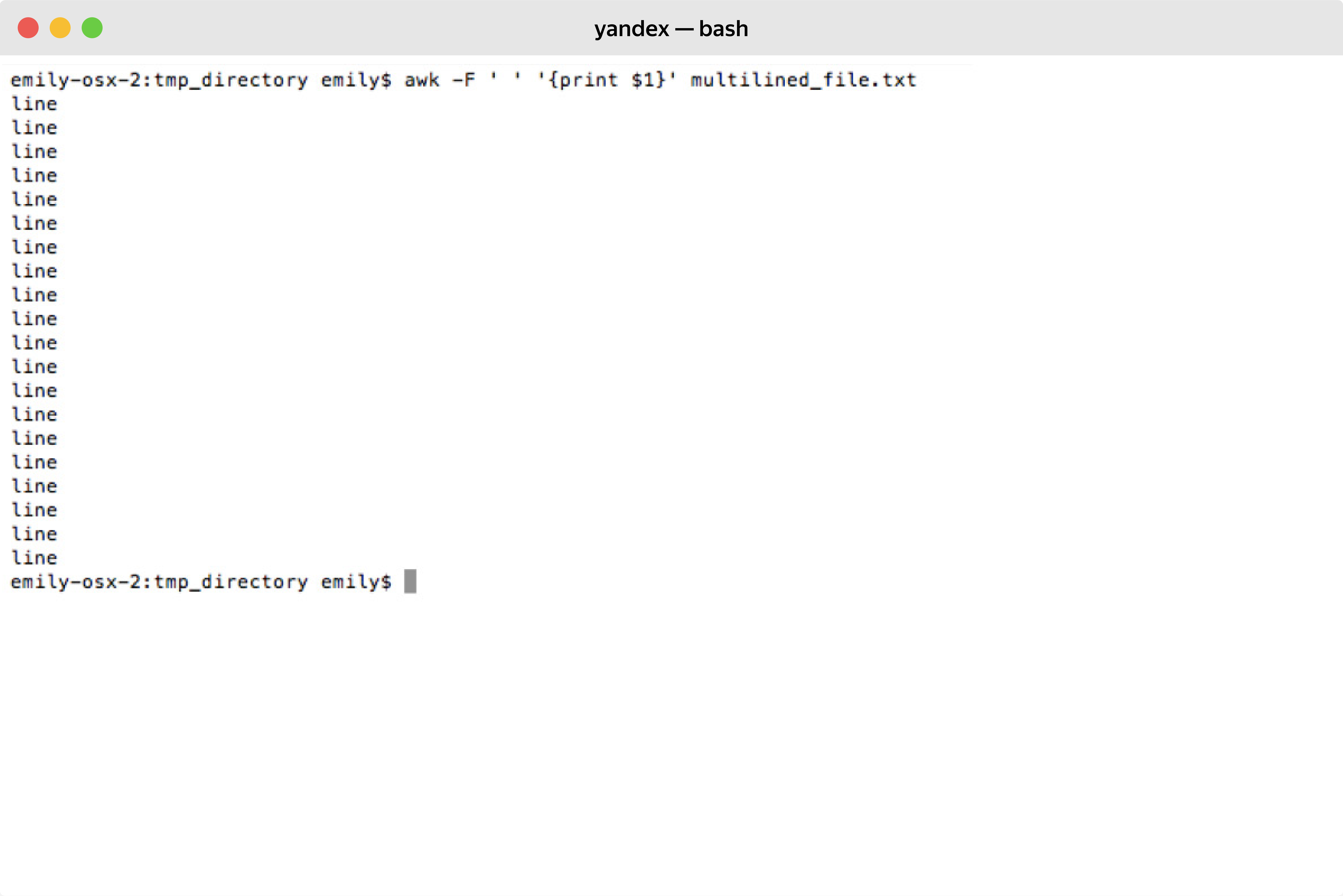
```
emily-osx-2:tmp_directory emily$ sort -k2,2 -n -r multilined_file.txt
line 20
line 19
line 18
line 17
line 16
line 15
line 14
line 13
line 12
line 11
line 10
line 9
line 8
line 7
line 6
line 5
line 4
line 3
line 2
line 1
emily-osx-2:tmp_directory emily$ 
```

awk

- "Aho, Weinberger, and Kernighan”, Bell Labs, 1970s
- interpreted programming language text processing

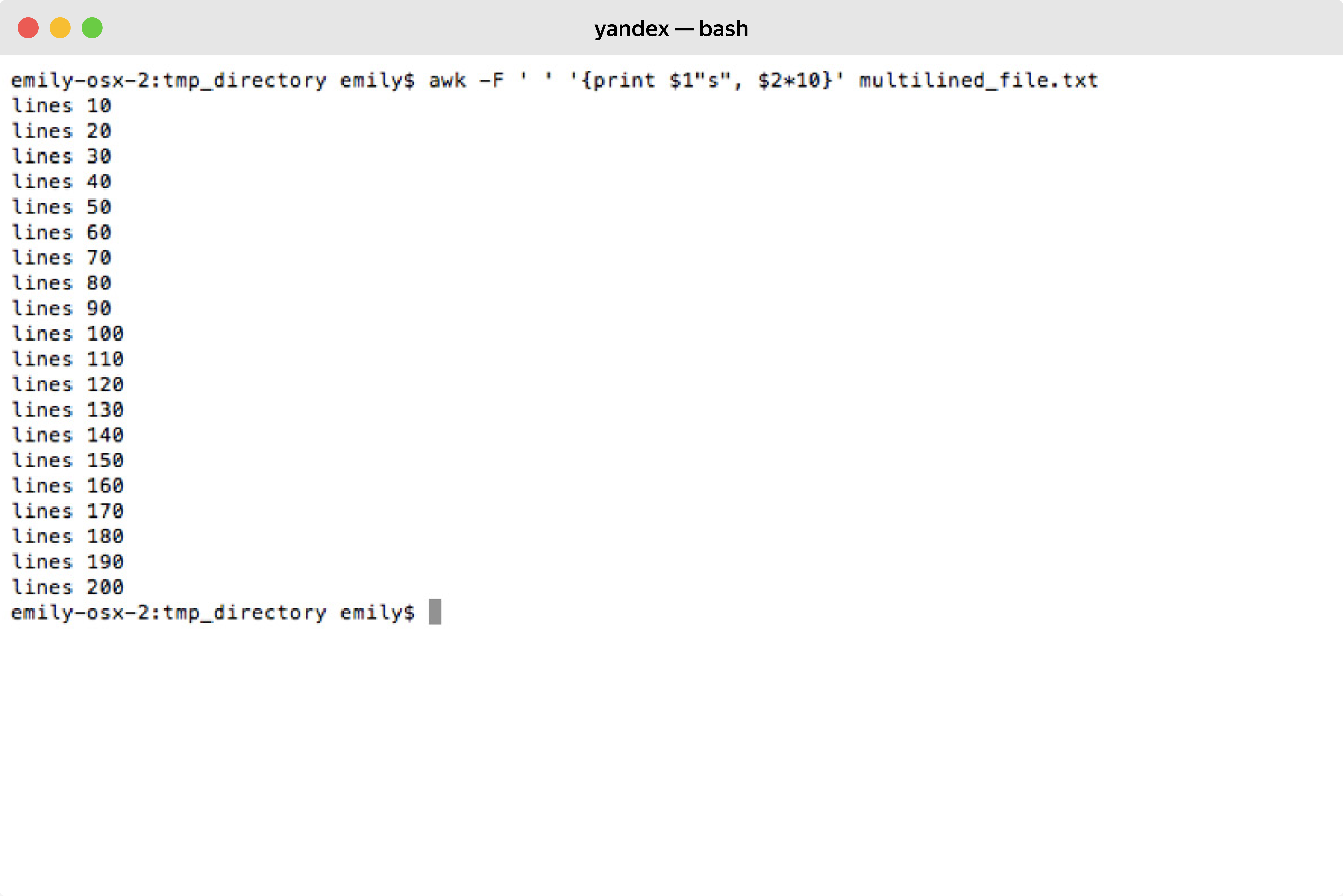
```
awk [ -F fs ] [ -v var=value ] [ 'prog' | -f progfile ] [ file ... ]
```

awk



```
emily-osx-2:tmp_directory emily$ awk -F ' ' '{print $1}' multilined_file.txt
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
line
emily-osx-2:tmp_directory emily$ 
```

awk

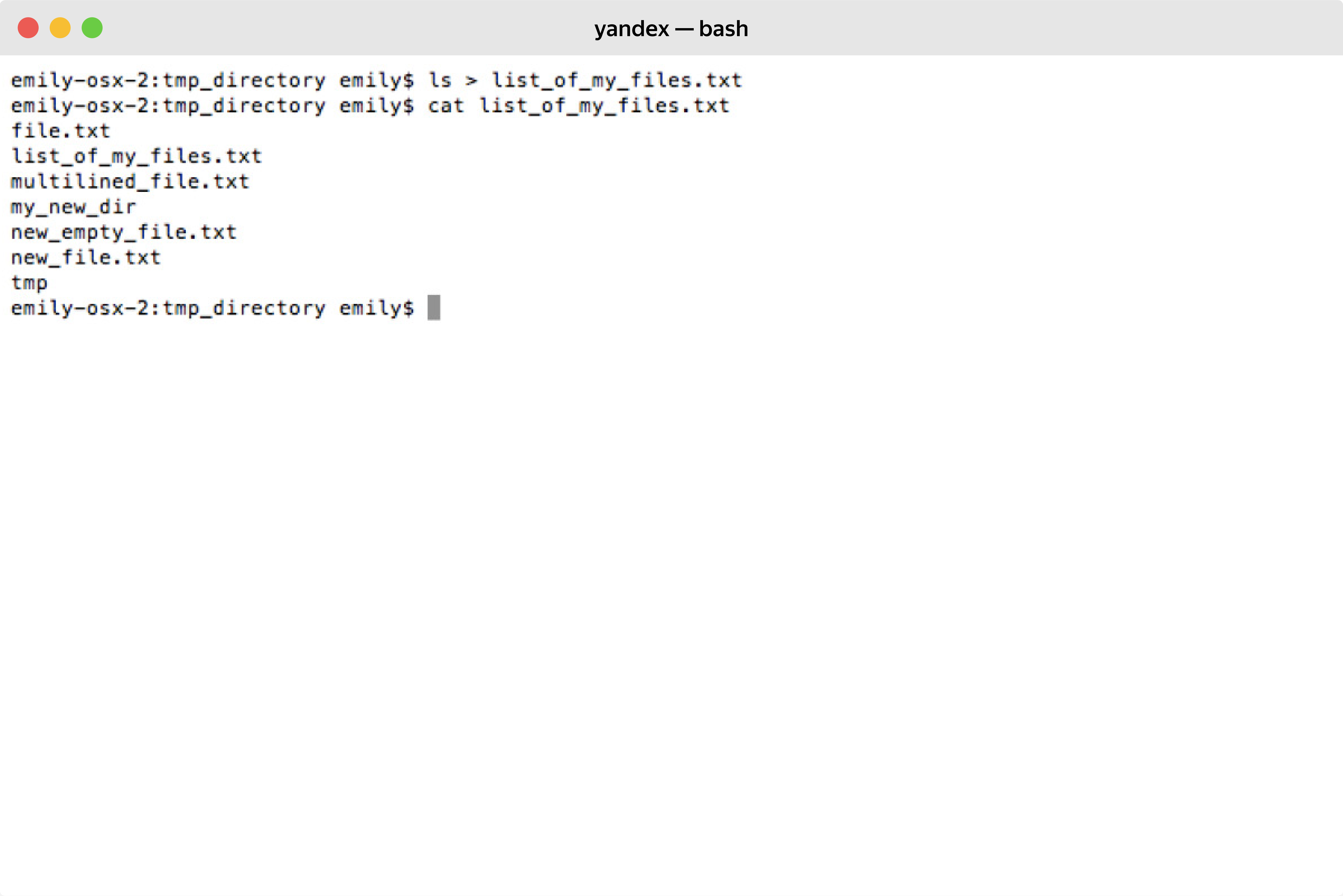


```
emily-osx-2:tmp_directory emily$ awk -F ' ' '{print $1"s", $2*10}' multilined_file.txt
lines 10
lines 20
lines 30
lines 40
lines 50
lines 60
lines 70
lines 80
lines 90
lines 100
lines 110
lines 120
lines 130
lines 140
lines 150
lines 160
lines 170
lines 180
lines 190
lines 200
emily-osx-2:tmp_directory emily$ 
```

operators

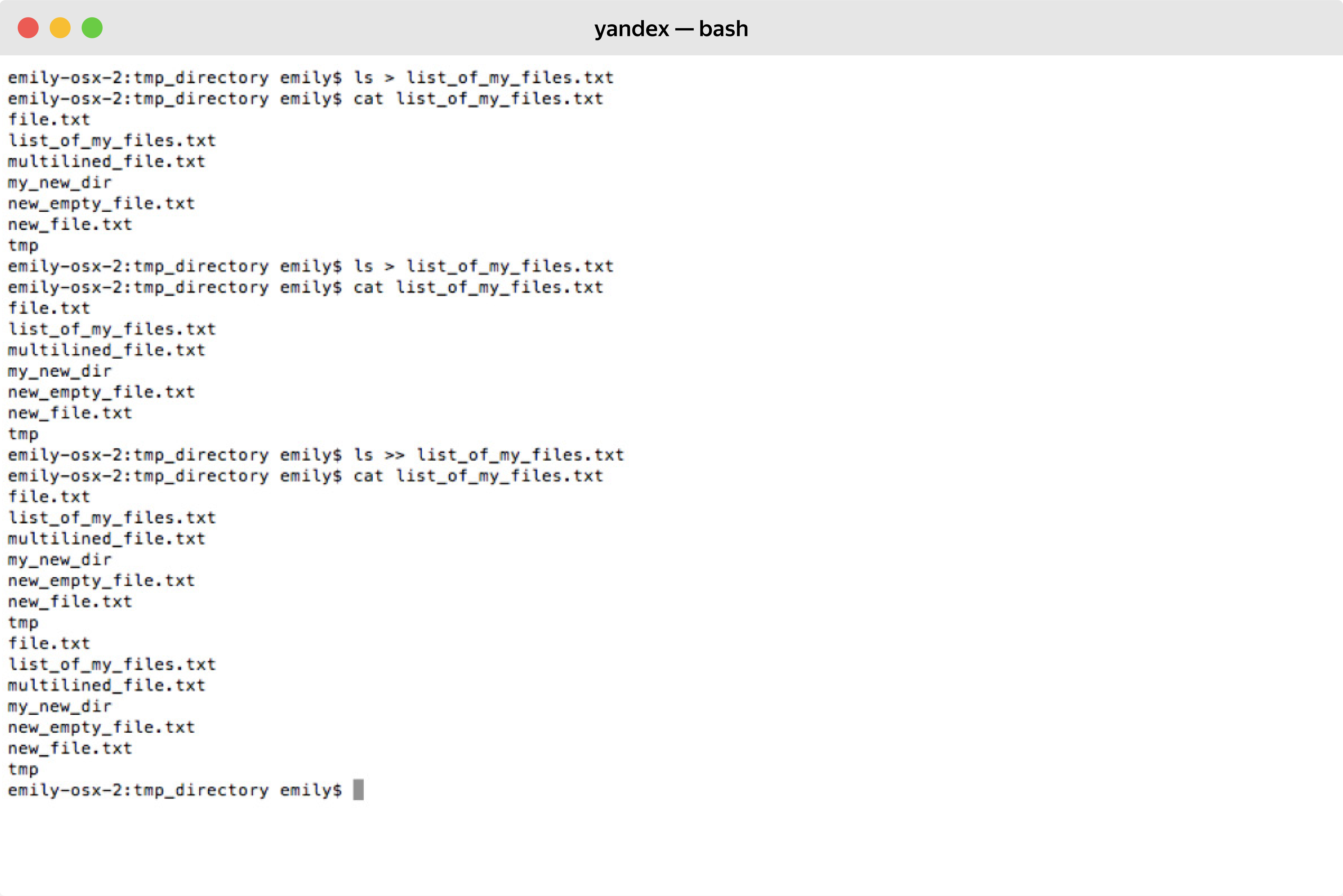
- > overwrites the file if it exists or creates it if it doesn't exist
- >> appends to a file or creates the file if it doesn't exist

> operator



```
emily-osx-2:tmp_directory emily$ ls > list_of_my_files.txt
emily-osx-2:tmp_directory emily$ cat list_of_my_files.txt
file.txt
list_of_my_files.txt
multilined_file.txt
my_new_dir
new_empty_file.txt
new_file.txt
tmp
emily-osx-2:tmp_directory emily$
```

>> operator

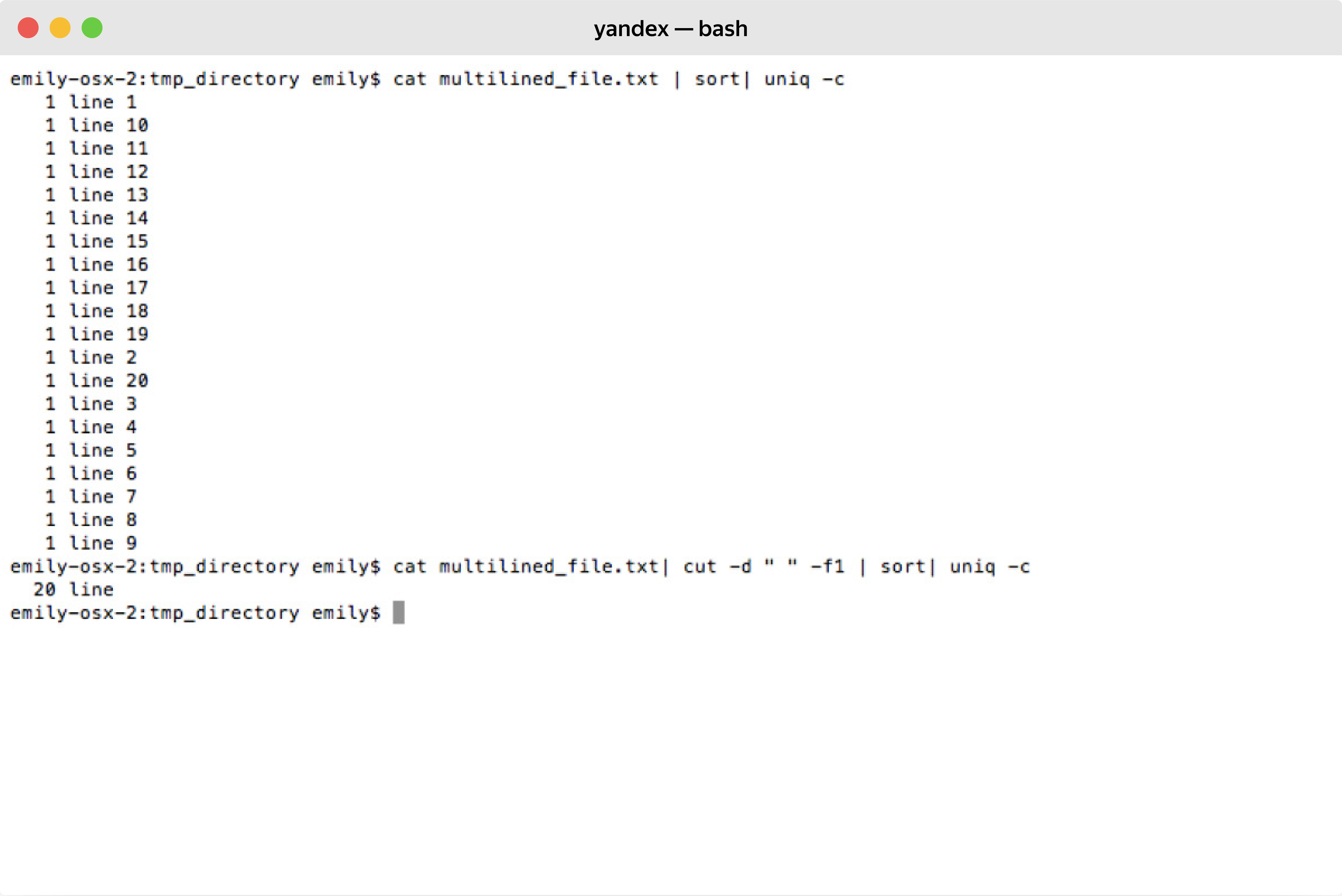


```
emily-osx-2:tmp_directory emily$ ls > list_of_my_files.txt
emily-osx-2:tmp_directory emily$ cat list_of_my_files.txt
file.txt
list_of_my_files.txt
multilined_file.txt
my_new_dir
new_empty_file.txt
new_file.txt
tmp
emily-osx-2:tmp_directory emily$ ls > list_of_my_files.txt
emily-osx-2:tmp_directory emily$ cat list_of_my_files.txt
file.txt
list_of_my_files.txt
multilined_file.txt
my_new_dir
new_empty_file.txt
new_file.txt
tmp
emily-osx-2:tmp_directory emily$ ls >> list_of_my_files.txt
emily-osx-2:tmp_directory emily$ cat list_of_my_files.txt
file.txt
list_of_my_files.txt
multilined_file.txt
my_new_dir
new_empty_file.txt
new_file.txt
tmp
file.txt
list_of_my_files.txt
multilined_file.txt
my_new_dir
new_empty_file.txt
new_file.txt
tmp
emily-osx-2:tmp_directory emily$ 
```

operators

- | pipe operator, output of first command acts as an input to the second command

pipe



```
emily-osx-2:tmp_directory emily$ cat multilined_file.txt | sort| uniq -c
   1 line 1
   1 line 10
   1 line 11
   1 line 12
   1 line 13
   1 line 14
   1 line 15
   1 line 16
   1 line 17
   1 line 18
   1 line 19
   1 line 2
   1 line 20
   1 line 3
   1 line 4
   1 line 5
   1 line 6
   1 line 7
   1 line 8
   1 line 9
emily-osx-2:tmp_directory emily$ cat multilined_file.txt| cut -d " " -f1 | sort| uniq -c
  20 line
emily-osx-2:tmp_directory emily$ 
```

More operators!

- & make the command run in background
- || execute the second command only, if the execution of first command succeeds
- && execute second command only if the execution of first command fails

In-video quiz

- How to quit vim redactor without saving the edited file?
- How to substitute all commas ‘,’ to spaces ‘ ’ in a file?
- How to take first 5 columns of file and write into another file?

Outcome

- We use the following list of commands for file editing and exploring:
  - **vim** — text editor
  - **cut** — drop sections of files line by line
  - **tr** — translate one substring to another
  - **sort** — sort file by lines
  - **awk** — programming language for text editing
  - **special symbolic:** |, ||, &, &&, >, >>

BigDATAteam