



## ☆ Large Responses



Consider a text file (i.e., a file with a `.txt` extension) where each line contains a single log record with the following columns (in order):

1. The *hostname* of the *host* making the request.
2. This column's values are missing and described by a hyphen (i.e., `-`).
3. This column's values are missing and described by a hyphen (i.e., `-`).
4. A timestamp enclosed in square brackets following the format `[DD/mmm/YYYY:HH:MM:SS -0400]`, where `DD` is the day of the month, `mmm` is the name of the month, `YYYY` is the year, `HH:MM:SS` is the time in 24-hour format, and `-0400` is the time zone.
5. The *request*, enclosed in quotes (e.g., `"GET /images/NASA-logosmall.gif HTTP/1.0"`).
6. The *HTTP response code*.
7. The total number of *bytes* sent in their response.

### ► Example

Given a string, *filename*, denoting the name of a real text file that has a `.txt` extension (e.g., *filename.txt*), create a file named `bytes_filename.txt` (where *filename* is the file name string) to store information about large responses. The created file must contain two lines:

1. The first line must contain the number of requests that have more than 5000 bytes sent in their response.
2. The second line must contain the total sum of bytes sent by all responses sending more than 5000 bytes.

### Input Format

The given code in the editor reads the string denoting *filename* from STDIN.

### Constraints

- It is guaranteed that the total number of bytes sent by all large responses does not exceed  $10^{12}$ .
- The log file contains no more than  $2 \times 10^5$  records.

### Output Format

Create a file named `bytes_filename.txt` (where *filename* is the file name string) to store information about large responses. The created file must contain two lines:



more than 5000 bytes.

### Sample Input

```
hosts_access_log_00
```

### Sample Output

Given *filename* = "hosts\_access\_log\_00", we process the records in hosts\_access\_log\_00.txt and create an output file named bytes\_hosts\_access\_log\_00.txt containing the following rows:

```
2
80620
```

### Explanation

The log file hosts\_access\_log\_00.txt contains the following log records:

```
unicomp6.unicomp.net - - [01/Jul/1995:00:00:06 -0400] "GET
/shuttle/countdown/ HTTP/1.0" 200 3985
burger.letters.com - - [01/Jul/1995:00:00:11 -0400] "GET
/shuttle/countdown/liftoff.html HTTP/1.0" 304 0
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET /images/NASA-
logosmall.gif HTTP/1.0" 304 0
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET
/shuttle/countdown/video/livevideo.gif HTTP/1.0" 200 0
d104.aa.net - - [01/Jul/1995:00:00:13 -0400] "GET /shuttle/countdown/
HTTP/1.0" 200 3985
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET
/shuttle/countdown/count.gif HTTP/1.0" 200 40310
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /images/NASA-
logosmall.gif HTTP/1.0" 200 786
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /images/KSC-
logosmall.gif HTTP/1.0" 200 1204
d104.aa.net - - [01/Jul/1995:00:00:15 -0400] "GET
/shuttle/countdown/count.gif HTTP/1.0" 200 40310
d104.aa.net - - [01/Jul/1995:00:00:15 -0400] "GET /images/NASA-
logosmall.gif HTTP/1.0" 200 786
```

When we review the data above, we find the following large responses:

1. The sixth log record sent a large response:

Hostname	-	-	Timestamp	Request	HTTP Response Code	Bytes
----------	---	---	-----------	---------	--------------------	-------



5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

-0400]

ntdown  
/count  
.gif  
HTTP/1  
.0"

Observe that  $40310 > 5000$ .

2. The ninth log record sent a large response:

Hostname	-	-	Timestamp	Request	HTTP Response Code	Bytes
d104.aa. net	-	-	[01/Jul/1 995:00:00 :15 -0400]	"GET /shutt le/cou ntdown /count .gif HTTP/1 .0"	200	4031 0

Observe that  $40310 > 5000$ .

Because there are a total of two records that sent a total of  $40310 + 40310 = 80620$  bytes in their collective responses, the first line of our output file is 2 and the second line of our output file is 80620.

We recommend you take a quick tour of our editor before you proceed.  
The timer will pause up to 90 seconds for the tour.

[Start tour](#)

For help on how to read input and write output in 'Java 7', click here.

[Original code](#)

Java 7



```
1 ► import ↔;
```

```
4
```

```
5 ▼ public class Solution {  
6     private static final Scanner scan = new Scanner(System.in);  
7
```



```
11 filename = scan.nextLine(),
12
13     }
14 }
```

15

16

Line: 12 Col: 1

Run Code

Submit code & Continue

(You can submit any number of times)

☐ Test against custom input

 [Download sample test cases](#)

*The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*

[About](#) [Privacy Policy](#) [Terms of Service](#)