

Task #1



Unix command 'wc' takes a text file as input and tells you how many characters, words, and lines it has. A team is working on an improved version that adds a new function: it counts words and their frequency. I.e.: How many times each word appears in the text.

Example: given the input 'This is a line\nAnd this is another',

The new 'wc' provides the following output:

- lines = 2
- words = 8
- frequency = 'this' (2), 'is' (2), 'a' (1), 'line'(1), 'and' (1), another (1)

As the list of frequencies can be too long, it is determined that the first 5 more frequent words will be displayed.

If there are less than 5 words, the existing ones will be displayed.

According to the changes mentioned above, the frequency value is renamed as 'top-5', thus changing the output of the prior example as follows:

- lines = 2
- words = 8
- top-5 = 'is' (2), 'this' (2), 'a' (1), 'and' (1), 'line'(1)



- a. Detail of test cases (inputs) that would be generated by testing top-5 functionality
- b. Mention any assumptions you made or any information you feel is missing.

Assumptions:

File properties:


- **supported files extensions:** .txt, .log, .csv, .html
- **max file name:** 300 characters
- **max file size:** 1 GB
- **amount of files:** wc can read only one file. If multiple files are provided, an error will be thrown

Text properties:


- **case sensitive:** false
- **sorting of top-5:** first by frequency, secondly - alphabetically
- **punctuation (: ! , . etc):** ignored when counting words
- **max amount of characters:** 1000
- **supported encoding:** UTF-8
- **words delimiters:** spaces, \t, \n , Special characters(- _ /)
- **other special characters:** ignored when counting words
- **numbers:** ignored when counting words
- **empty lines:** ignored, not counted


Test data description:

I used the parameters mentioned below as "sanity parameters" if they are not important for a specific test, if the program doesn't work with these parameters, there is no point in testing edge cases. If the program passes the sanity testing, we can proceed to test various settings and combinations of the properties we are interested in.

Aa File content	≡ Words amount	≡ Lines	≡ Extension	≡ Unique words
Positive cases: 				
Sanity test: Small text input, small amount of words + popular extension + spaces as a delimiters + absolute path	(5, 100)	[2,500]	.txt	(5, 500]

Test Data top5: files and file content

Aa File content	≡ Words amount	≡ Lines	≡ Extension	≡ Unique words
Positive cases: 				
Sanity test: Small text input, small amount of words + popular extension + spaces as a delimiters + absolute path	(5, 100)	[2,500]	.txt	(5, 500]
one word [3-5 characters], .csv extension file in another folder with valid Relative path	1	1	.csv	1
one big word [1000 characters]	1	1	.log	1
exactly 5 unique words with same frequency, one line, .log extension, check top5 sorting	5	1	.log	5
exactly 5 unique words with different frequency check top5 sorting	(10,~)	[2,500]	any supported	5
> 5 unique words, big words amount check top5 sorting + all supported delimiters added: spaces, \t, \n, Special characters(- _/)	5000	any	.log	(5, ~)
less than 5 unique words + total words amount > 10 .html extension check top5 sorting	(10,~)	[2,500]	.html	(0,5)
mixed case: TEST, test, teSt	(5, 500]	[2,10]	any supported	(5, 500]
input where all 5 unique words repeated 1000 times and have long names (long top-5 line).	-	-	any supported	-
input contains all types of punctuation., ? ! : ; ,	(5, 500]	[2,10]	any supported	(5, 500]

Aa File content	≡ Words amount	≡ Lines	≡ Extension	≡ Unique words
<u>max amount of characters and lines: every word separated by new line character</u>	-		any supported	(5, 500]
<u>empty lines in file</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>empty file</u>	0	0	any supported	-
<u>file contains spaces only</u>	0	0	any supported	-
<u>file with a long name (300 char) and long path to file</u>	(5, 500]	[2,10]	any supported	-
<u>special characters and numbers “test” 77 te7fdf 2*3 *fff etc</u>	(5, 500]	[2,10]	.html	(5, 500]
<u>Untitled</u>				
<u>Error handling:</u> 				
<u>file with a too long name (> 301 char)</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>user doesn't have read permissions to provided file</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>unsupported file format</u>	any	any	.png, .exe	any
<u>corrupted file which is not possible to open</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>not supported encoding: non-UTF</u>	-	-	.txt	-
<u>big input (more than 1000 characters)</u>	-	[2,10]	any supported	-
<u>invalid path to file</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>more than 1 file is provided</u>	(5, 500]	[2,10]	any supported	(5, 500]
<u>command provided with unsupported option - -</u>	(5, 500]	[2,500]	any supported	
<u>big file that system cannot process</u>				

