**My Cat**

Remember to git add && git commit && git push each exercise!

We will execute your function with our test(s), please DO NOT PROVIDE ANY TEST(S) in your file

For each exercise, you will have to create a folder and in this folder, you will have additional files that contain your work. Folder names are provided at the beginning of each exercise under submit directory and specific file names for each exercise are also provided at the beginning of each exercise under submit file(s).

**My Cat**

* Submit directory: ex00
* Submit file: ["my\_cat.c"]

It's your turn to code the software cat!

Create a program called my\_cat which does the same thing as the system's cat command-line.

Read the content of each file which are received as an argument to your software. You don't have to handle any options. :-)

[[examples]] language = "Javascript"

$>node my\_cat file1

content\_file\_1

content\_file\_2

$>

**Example Python**

$>python my\_cat file1 file2

content\_file\_1

content\_file\_2

$>

**Example Ruby**

$>ruby my\_cat file1 file2

content\_file\_1

content\_file\_2

$>

**Example java**

$>java my\_cat file1 file2

content\_file\_1

content\_file\_2

$>

**Tips**

Google: man cat

Google: YOURCODINGLANGUAGE open file

Google: YOURCODINGLANGUAGE read file

Google: YOURCODINGLANGUAGE close file (don't forget to close.)

For javascript folks:

Google Why does Node.js' fs.readFile() return a buffer instead of string :-)