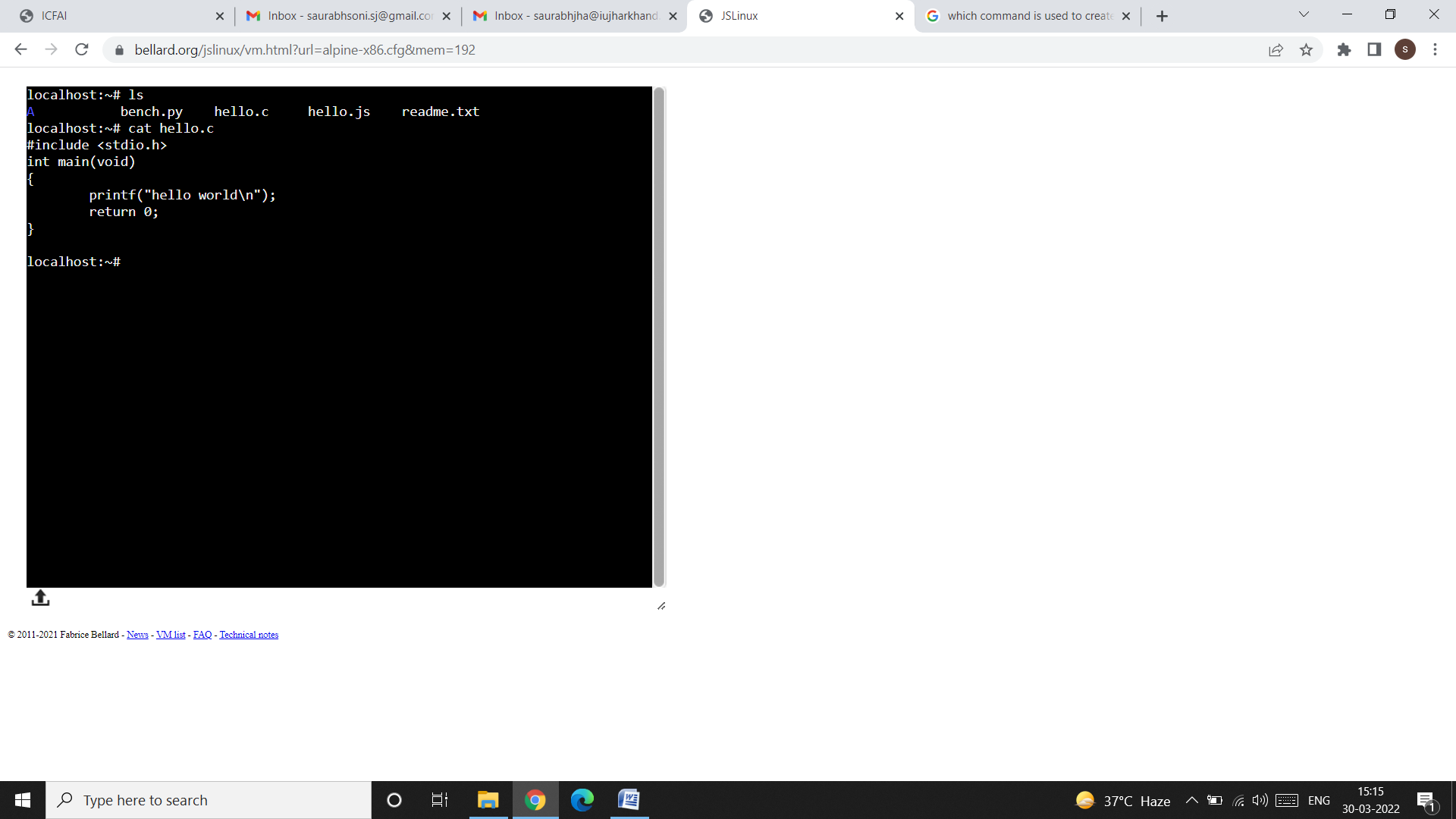
**cat (Displaying and Creating Files):**

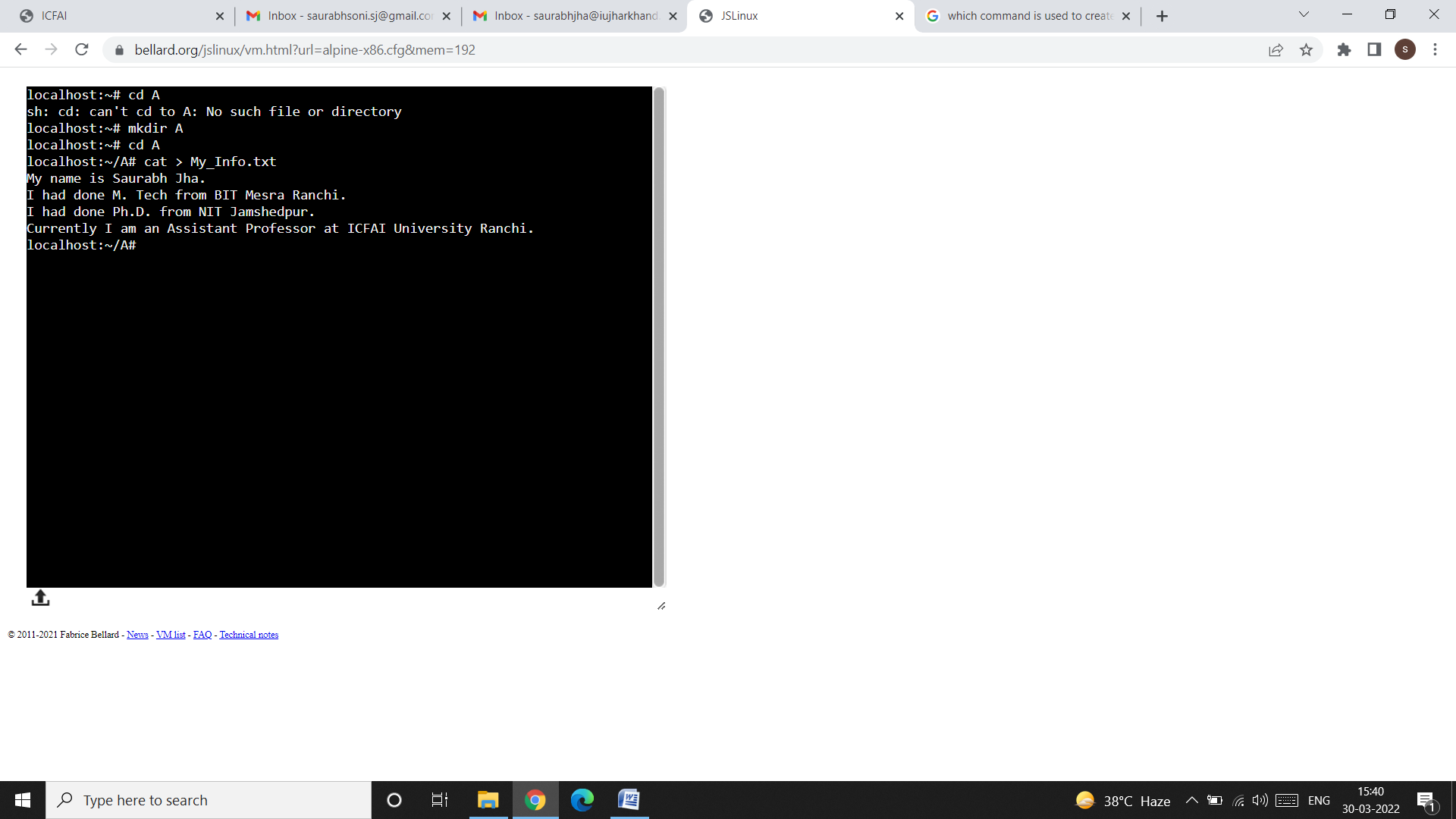
* The cat command is mainly used to display the contents of a small file on the terminal.



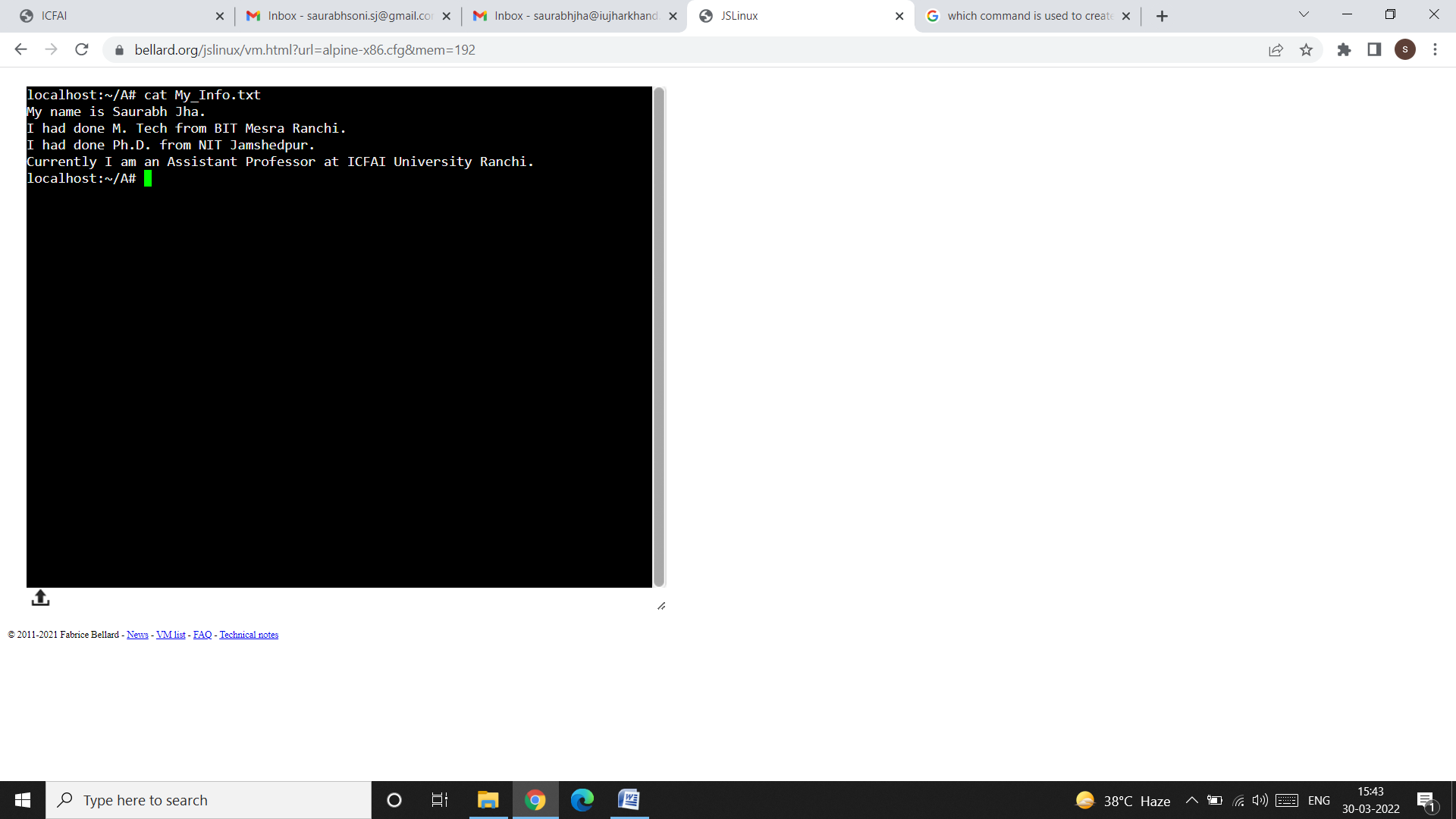
* cat also accepts more than one filename as arguments. The content of the second file will be shown immediately after the first file without any header information. In other words, cat concatenates the two files - hence its name.

**Using cat to create a file:**

* To create a file using cat command, enter the cat command followed by the > (the right chevron) character and the filename.
* When the command line is terminated with enter, the prompt vanishes. cat now waits to take input from the user. Write as many lines as you want and press Enter key after the end of every line. When you finish writing to the file, press ctrl and d key simultaneously to signify the end of input to the system.

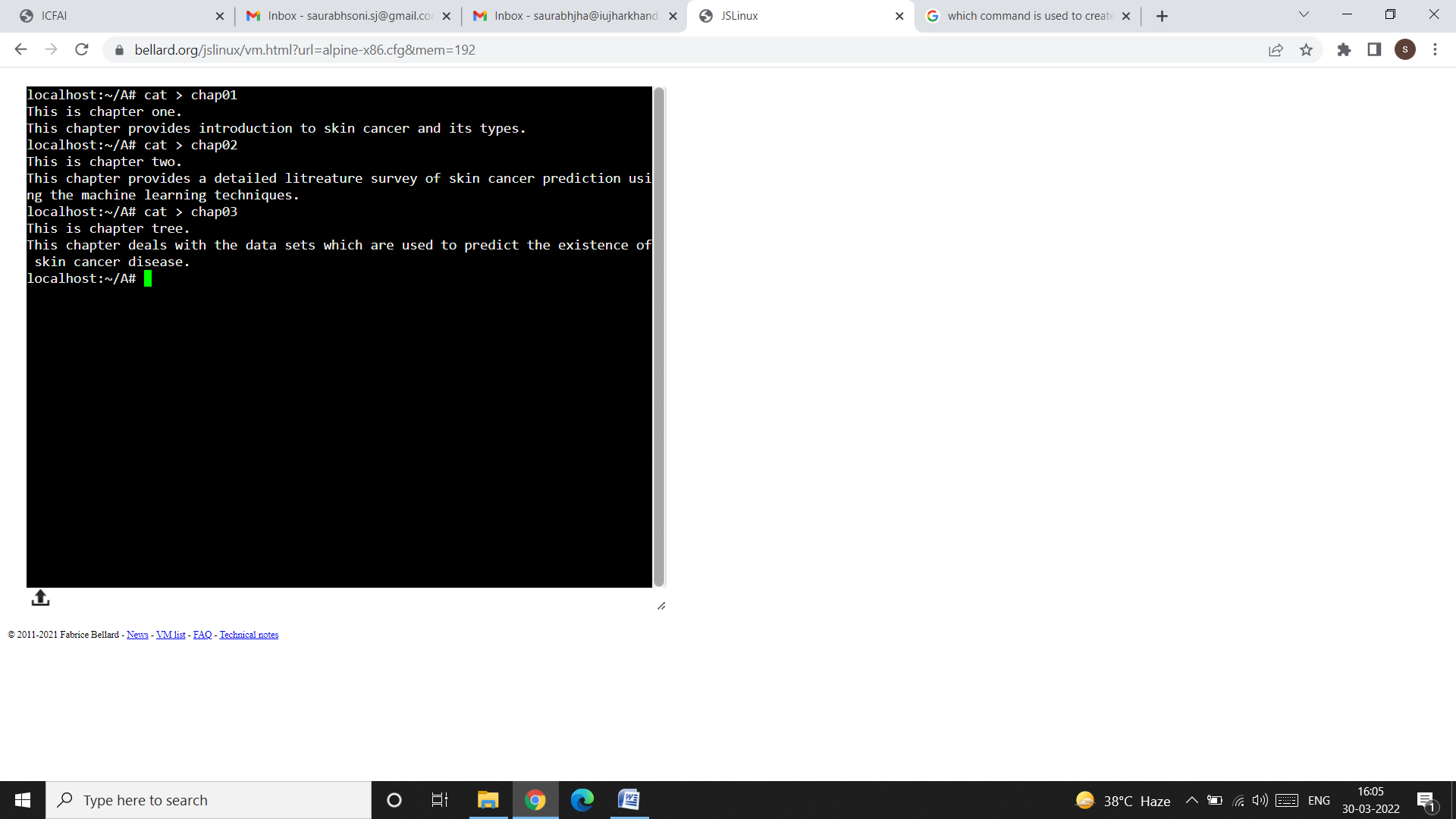


* In the above snap-shot, I have created a file named as My\_Info.txt in the directory A. We can create a file in any directory, even in the root directory too.
* To display the contents of the file which you have created, simply write the cat command followed by the filename like this:

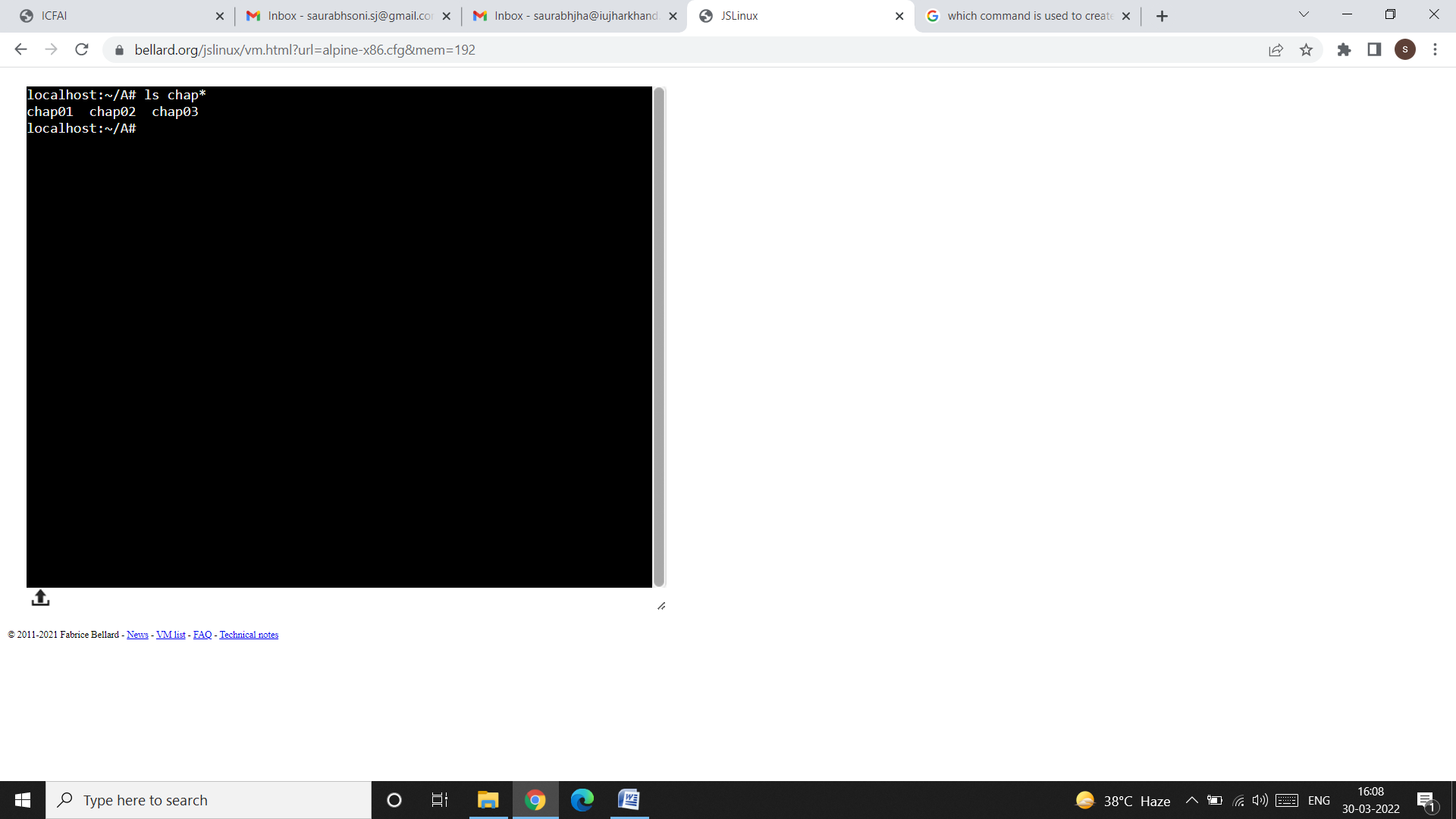


**The short-hand notation (\*):**

* The short-hand notation (\*) is used to access those files which has some of the characters common in the filename. For example, let us create three files in the directory A, with the filename as chap01, chap02, and chap03 like this:

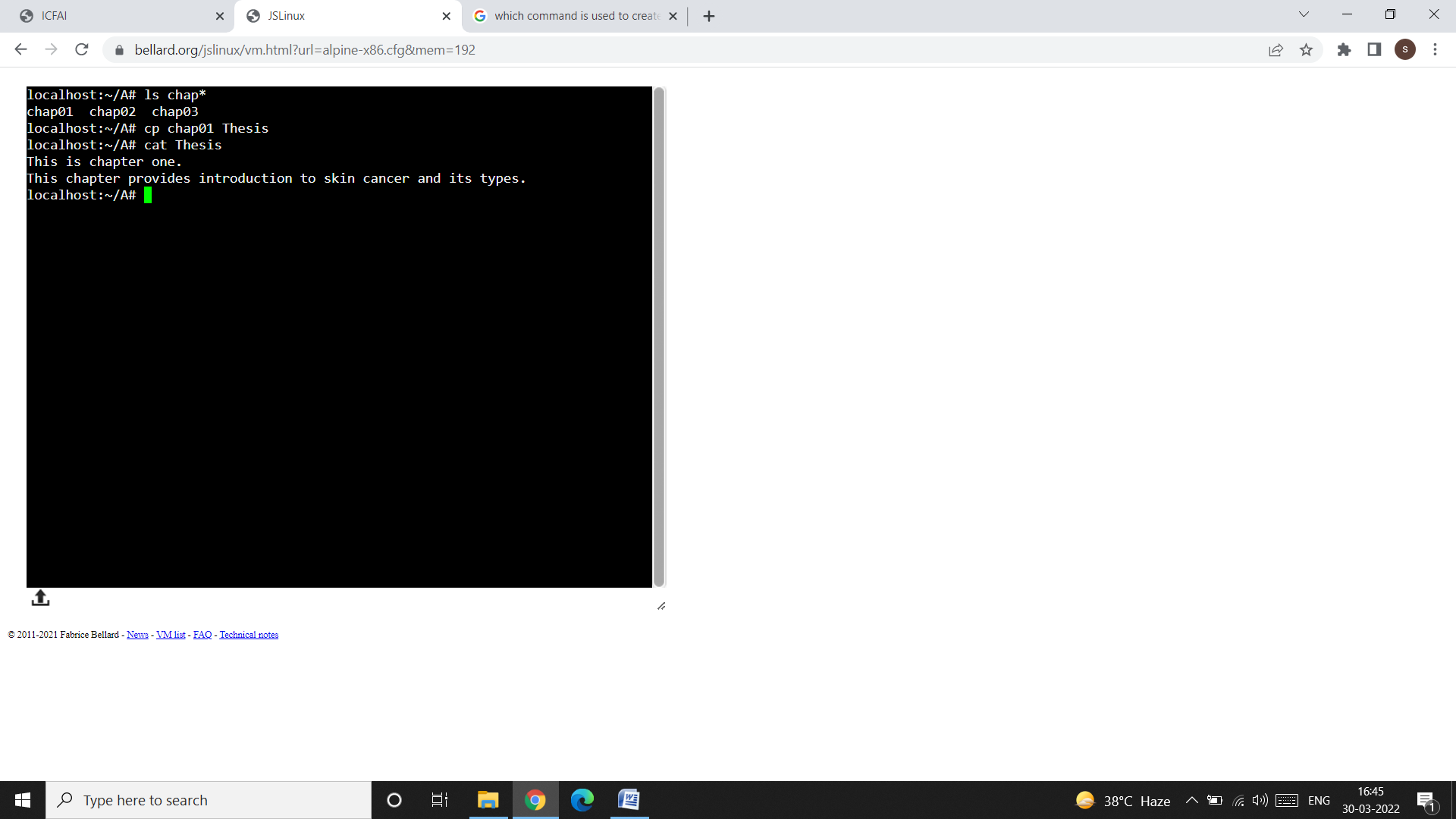


* With the help of the short-hand notation, we can get the detail of all the files starting with chap like this:

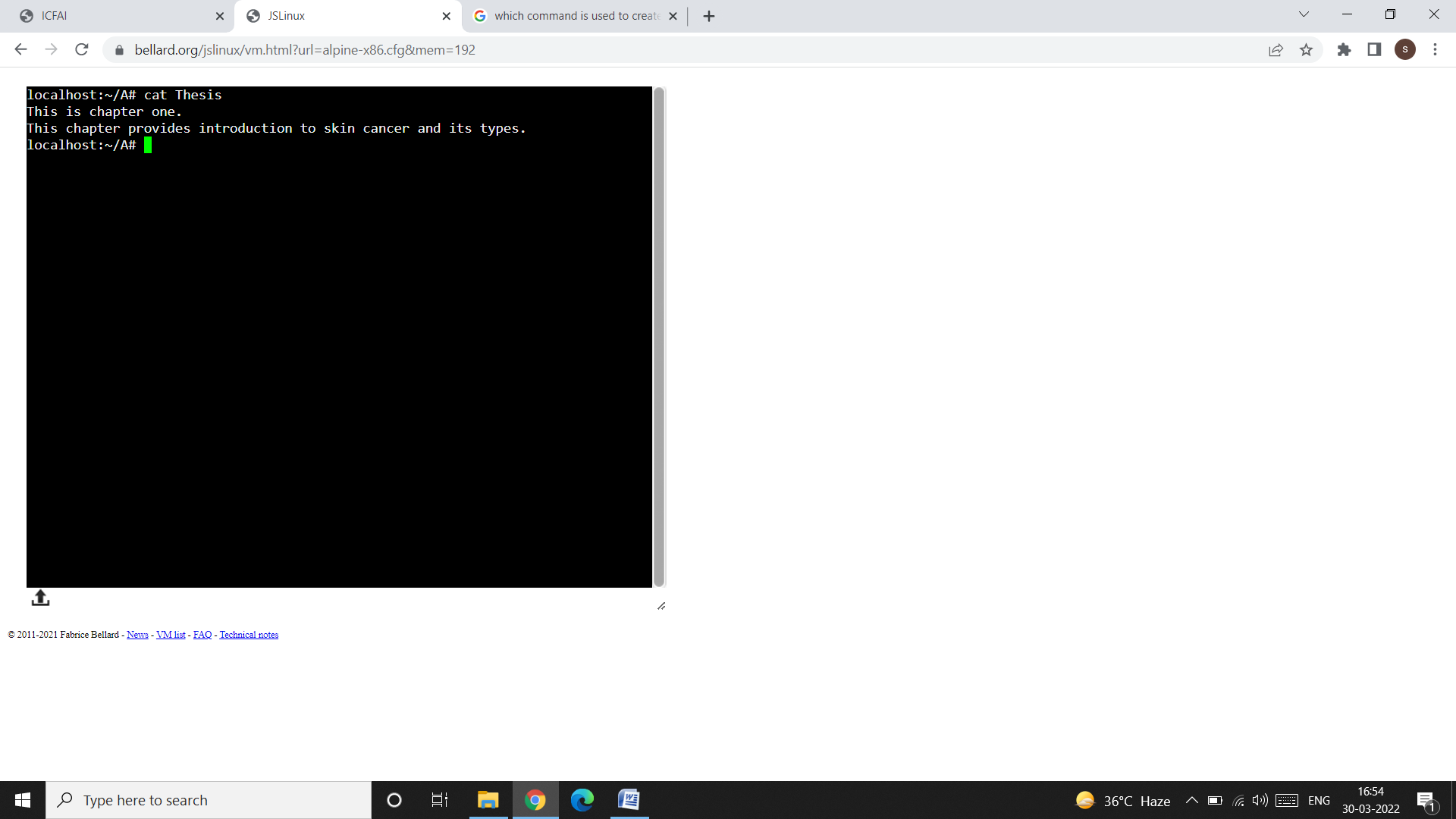


**cp (Copying a File):**

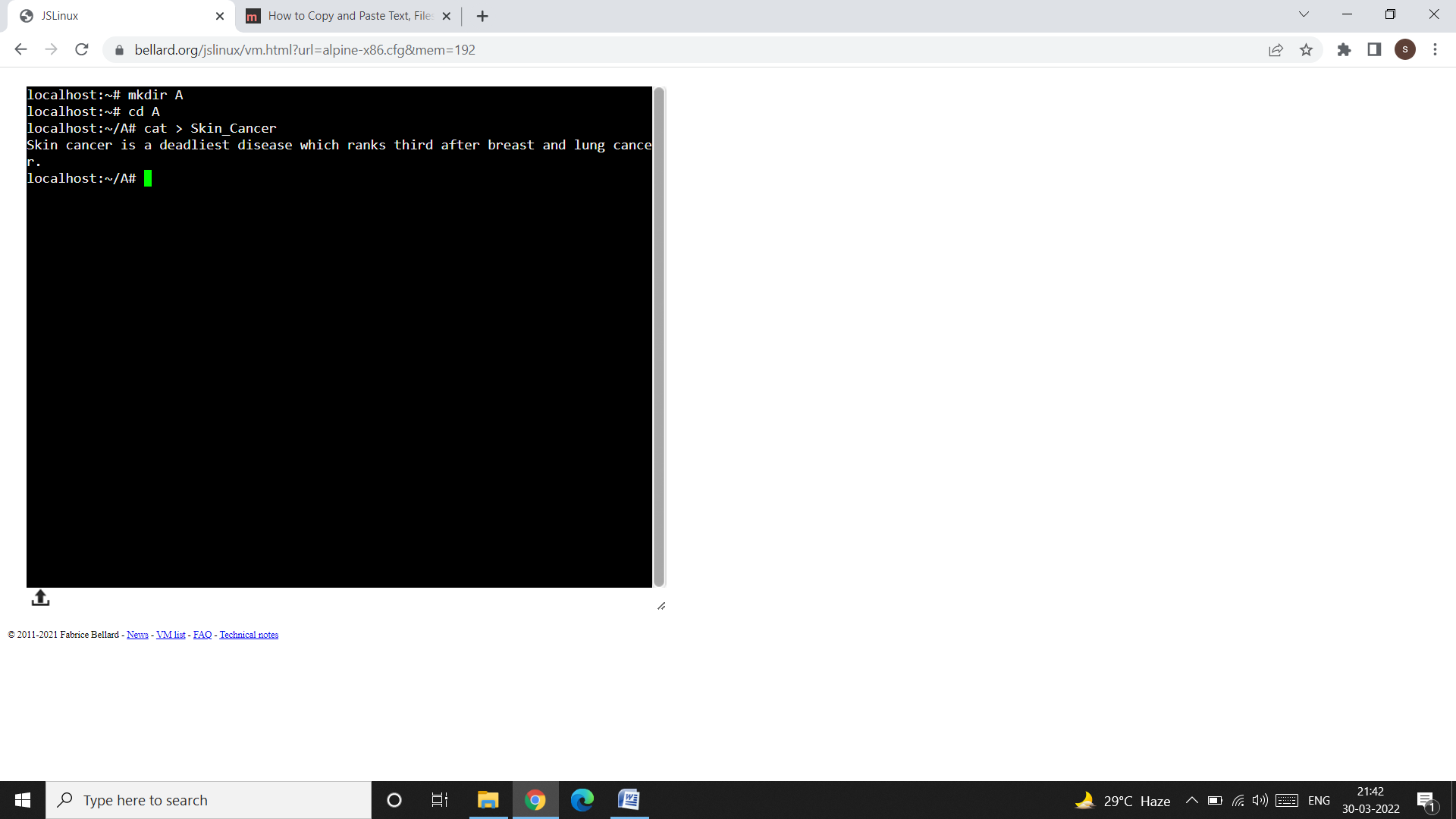
* The cp (copy) command copies a file or group of files. It creates an exact image of the file on disk with a different name.
* The syntax requires at least two filenames to be specified in the command line.
* When both are ordinary files, the first is copied to the second like this.



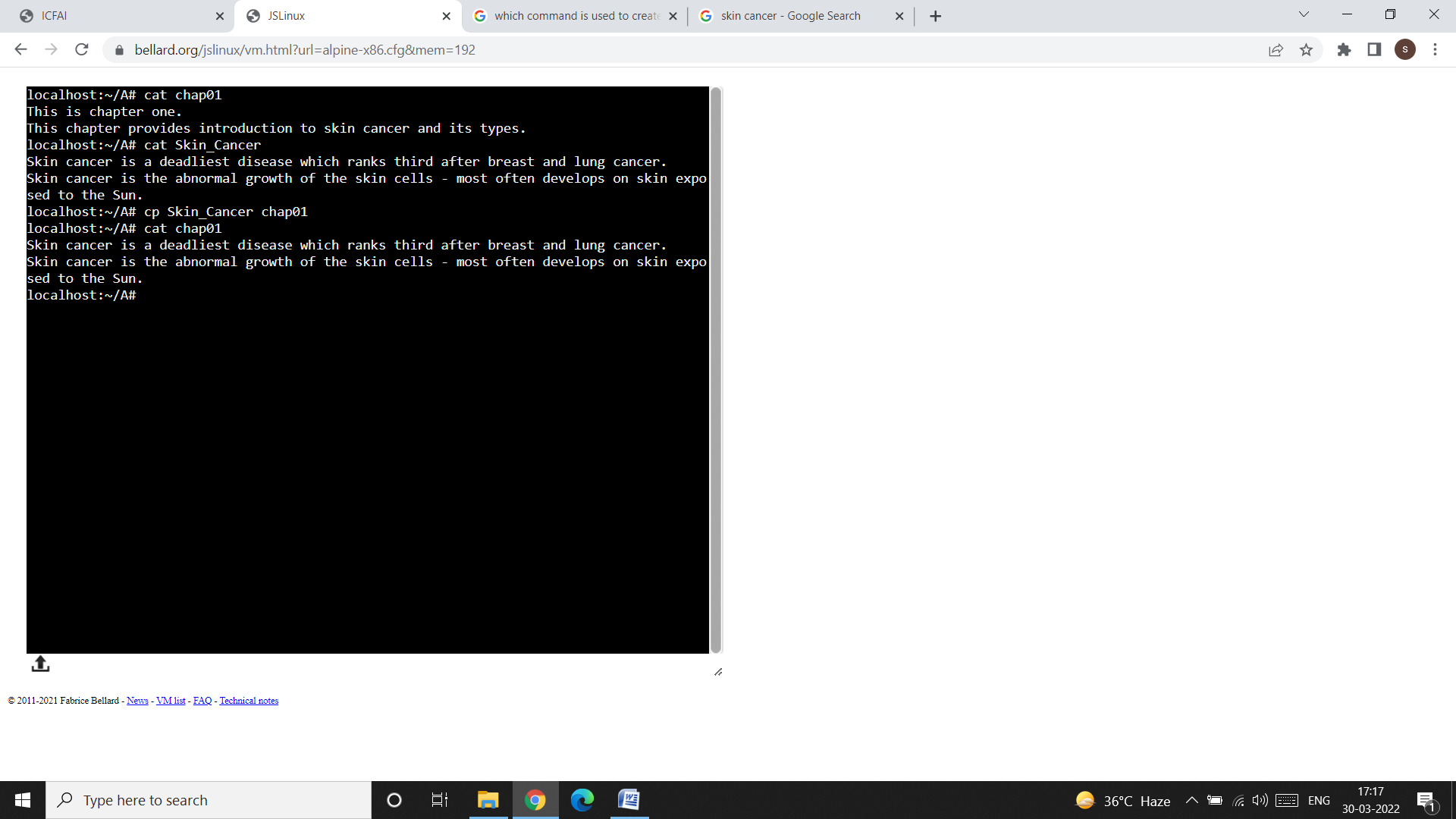
* In the above snap shot, first we have copied the contents of the file chap01 in the file Thesis. Note that, if the destination file (i.e. Thesis in our case) does not exist, it will first be created automatically by the system before copying takes place. Note that, we have never created the Thesis file in the directory A, but since it is used as a destination file for storing the contents of the file chap01, so the UNIX system automatically creates the Thesis file.
* Now, we can check the contents of the Thesis file using the cat command like this:



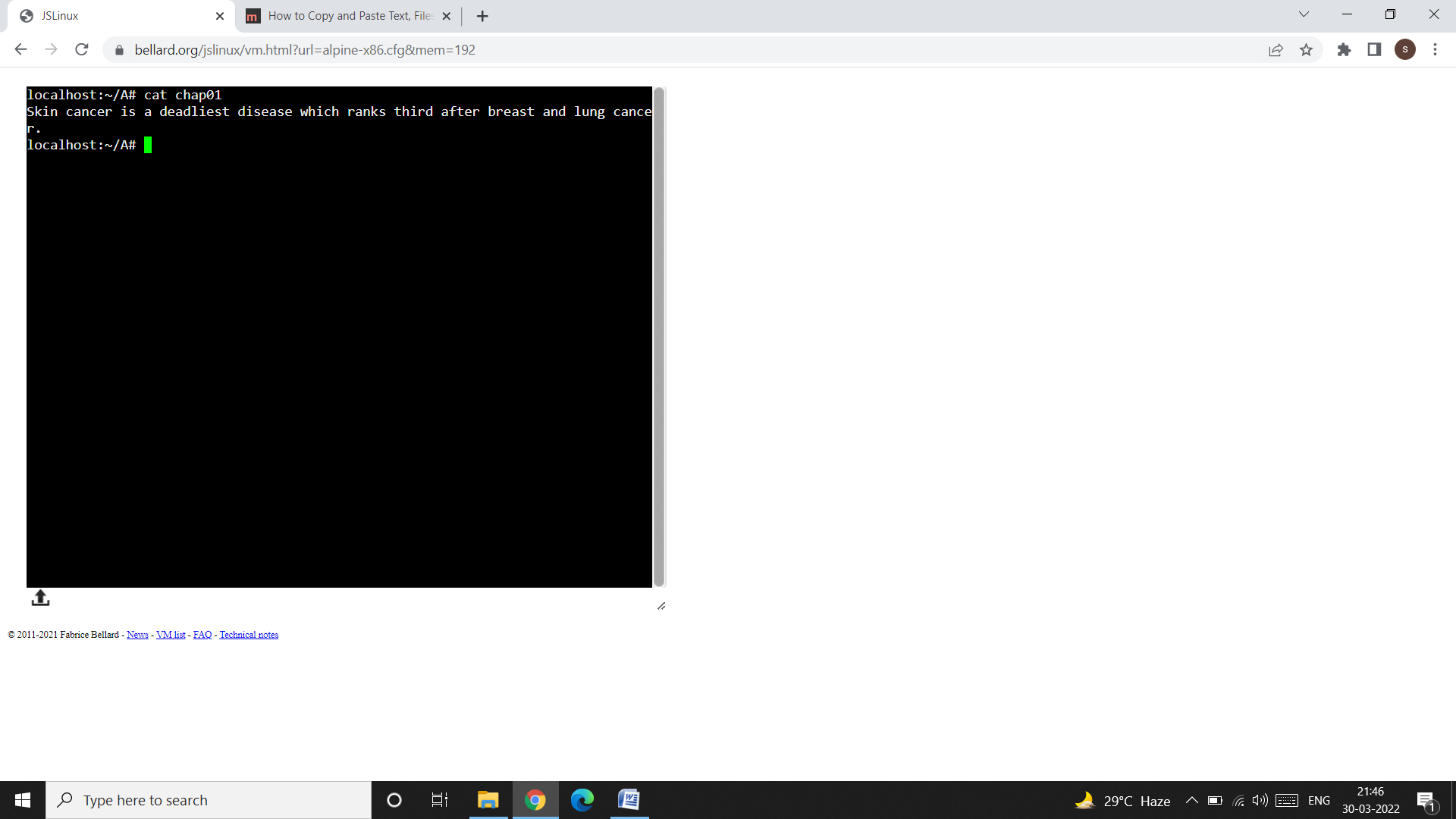
* From the above snap shot, we can see that the contents of the file chap01 are copied to the file Thesis.
* If there is only one file to be copied, the destination can be either an ordinary file or directory. You then have the option of choosing your destination filename.
* For example, let us create a file named as “Skin\_Cancer”, in the directory A first:



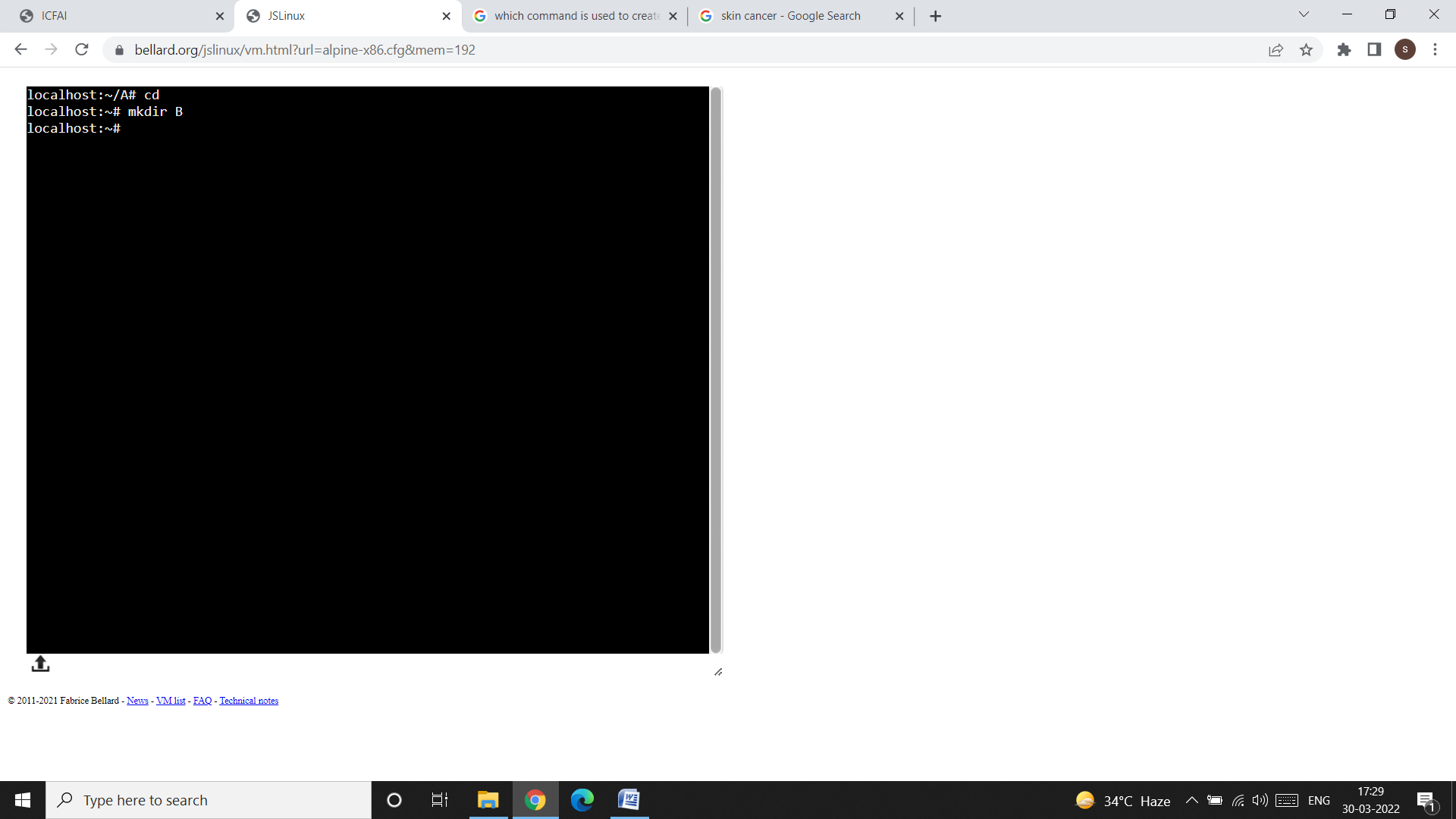
* Now, let us copy the contents of the file Skin\_Cancer to chap01 under the directory A like this:



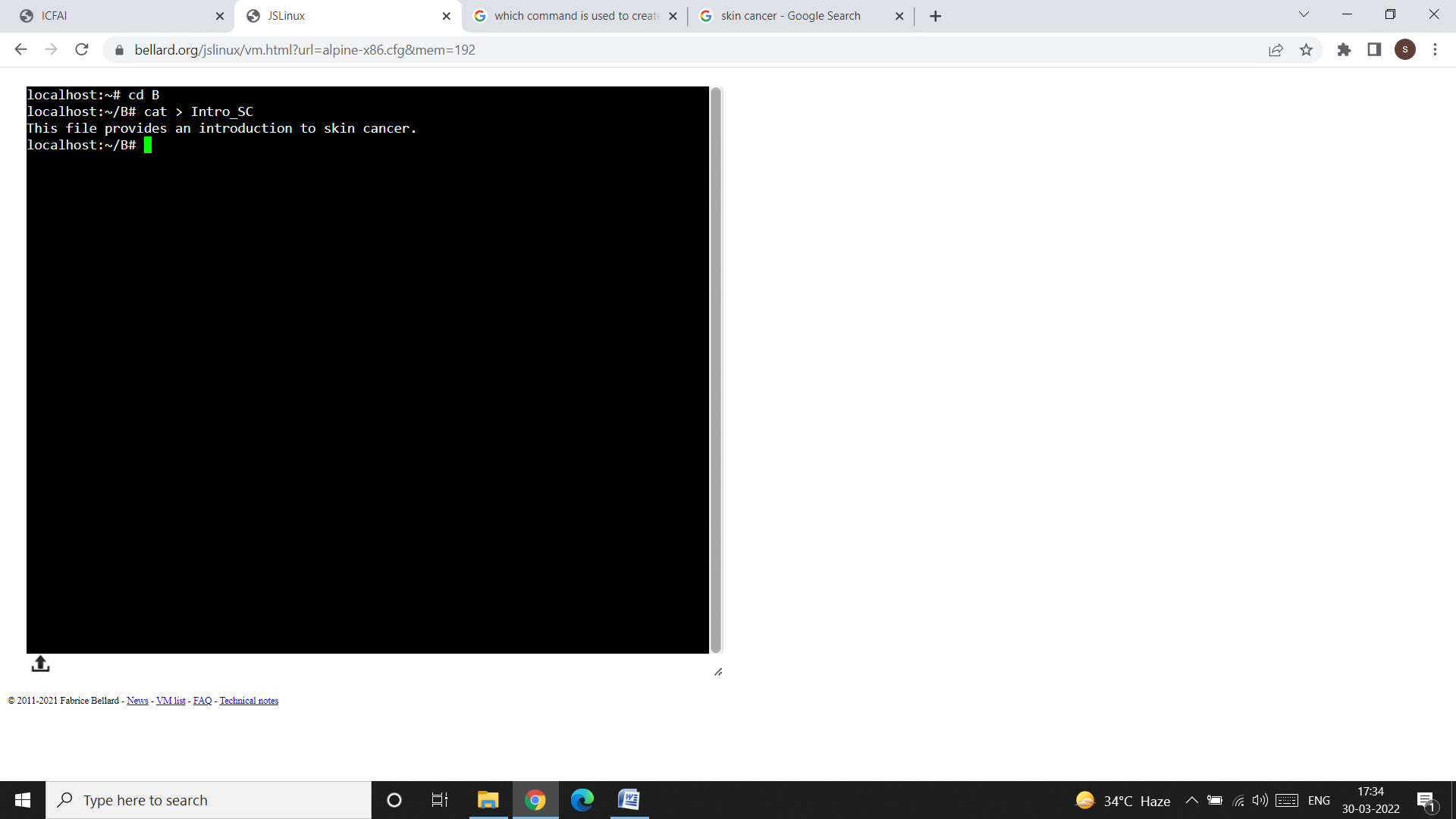
* Now, let us verify whether the contents of the file Skin Cancer has been copied to the file chap01 under directory A or not by using the cat command like this:



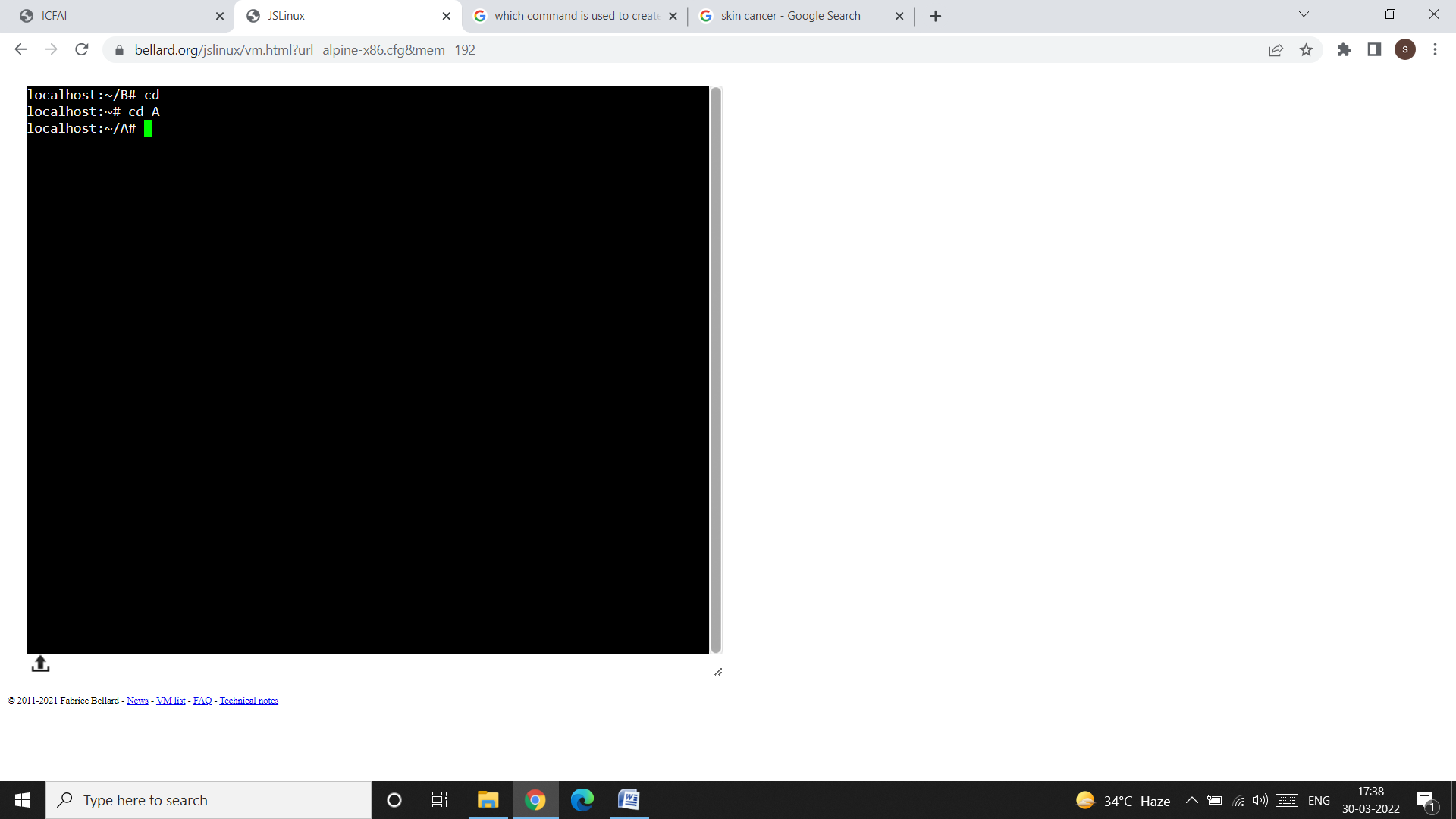
* **Note** that, the previous contents of chap01 (i.e., This is chapter one. This chapter provides introduction to skin cancer and its types.) is overwritten by using the cp command.
* Now let us create another directory named as “B”, in the root directory. To do so we have to exit from the directory A. For this, use the command cd and mkdir like this:



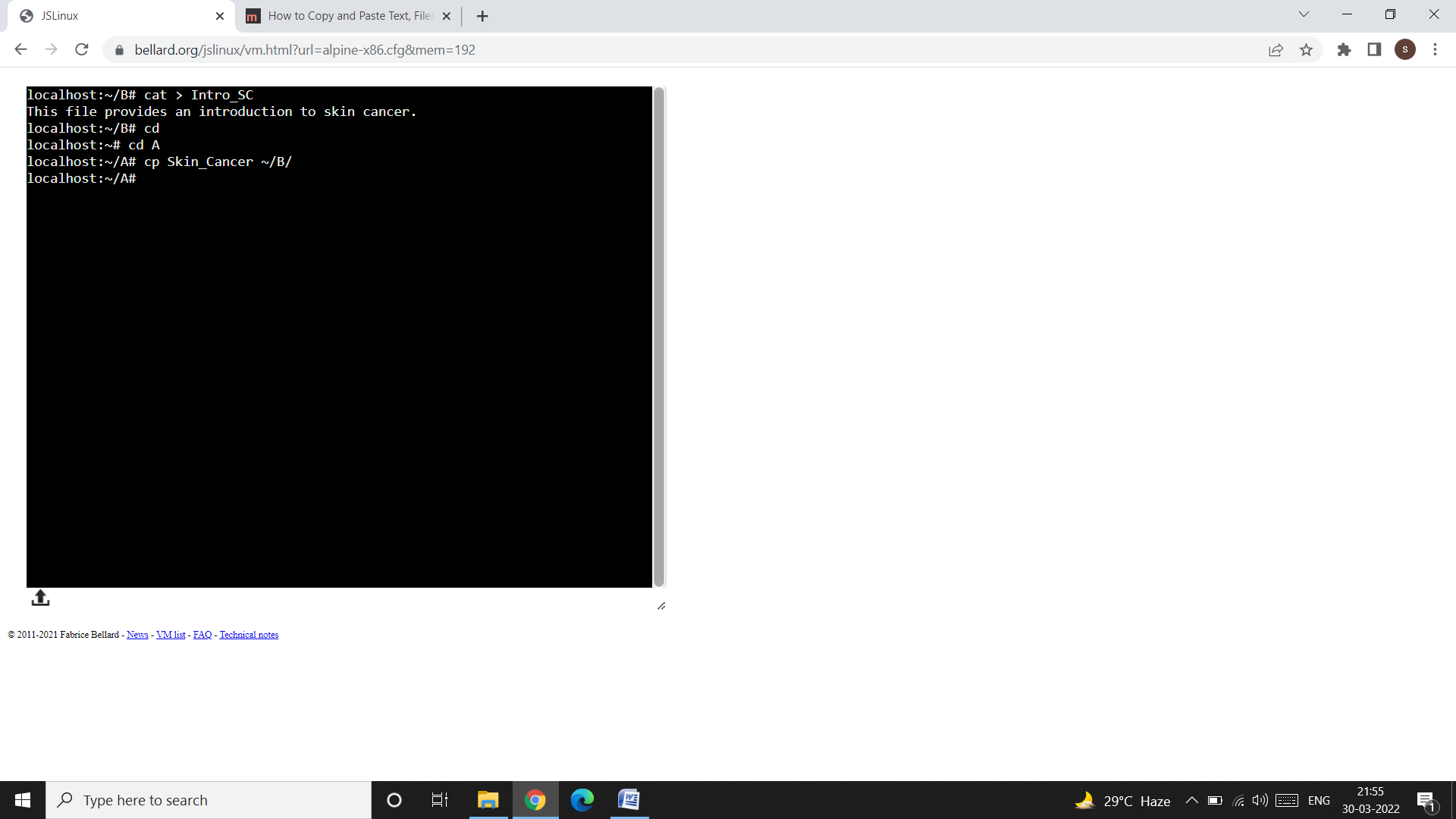
* Now, come to the directory B and create a file named as “Intro\_SC” like this:



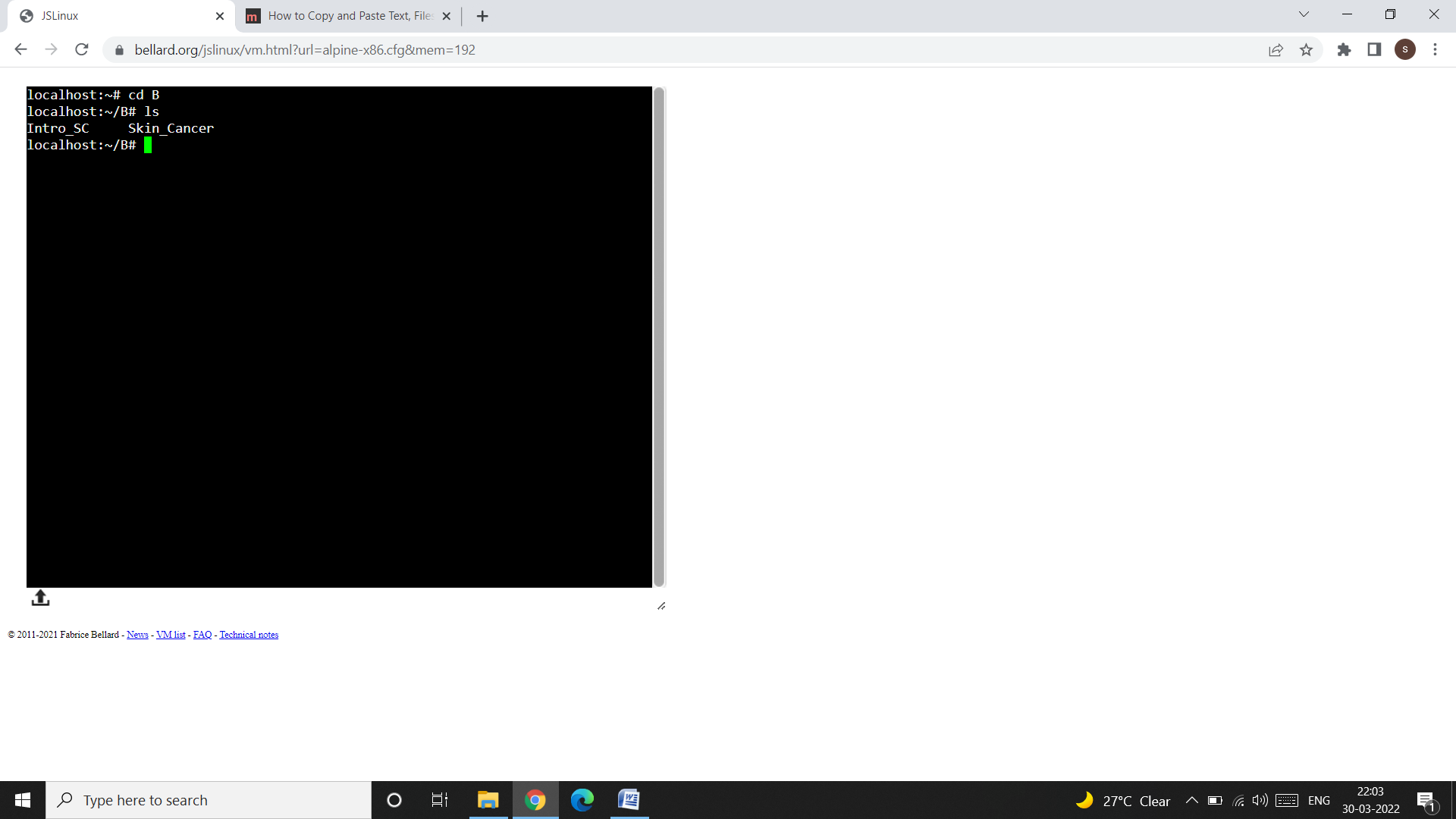
* Now, exit from directory B and enter in the directory A, like this:



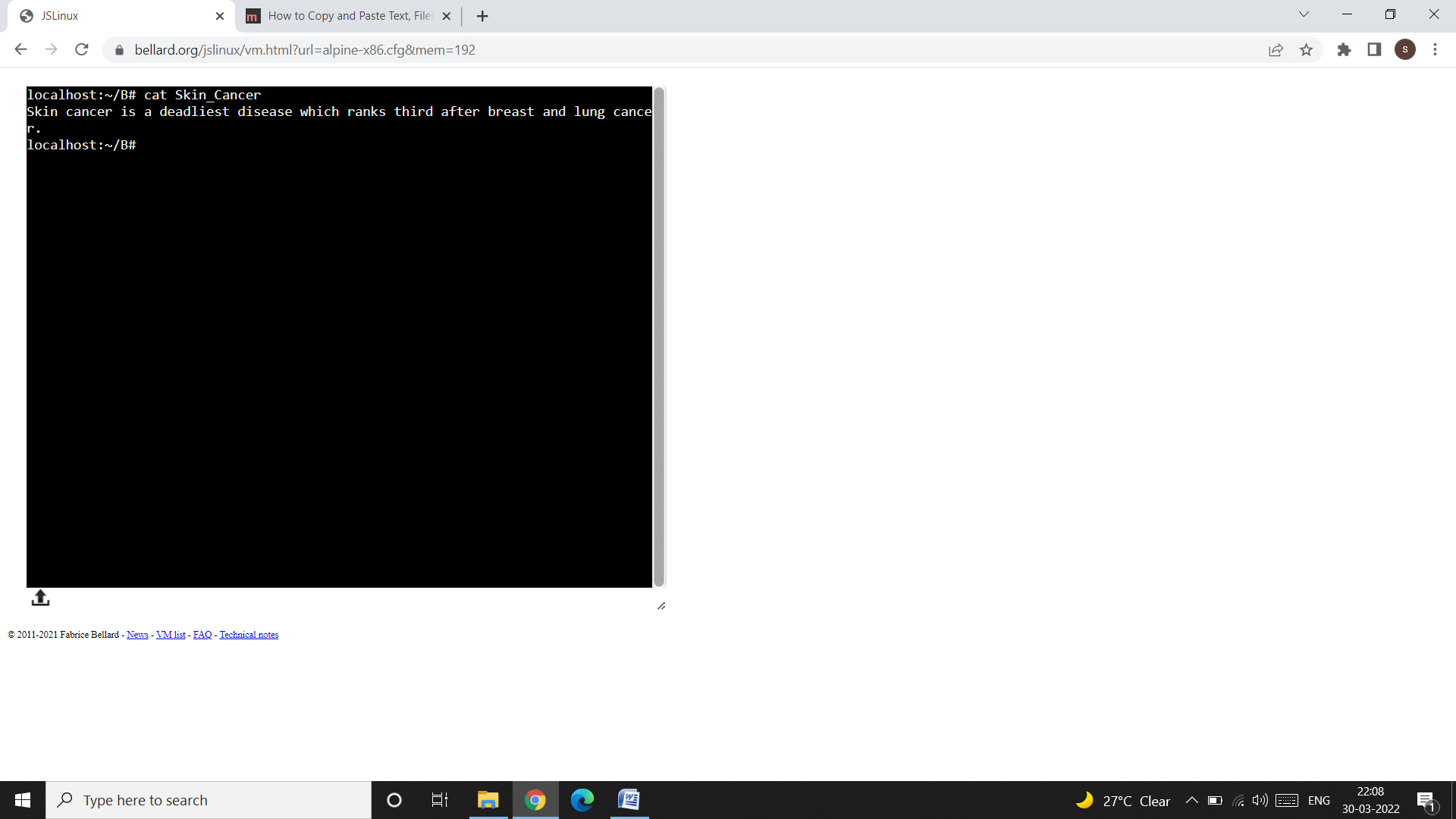
* Now, copy the contents of the file Skin\_Cancer (which is present in directory A) to the directory B, like this:

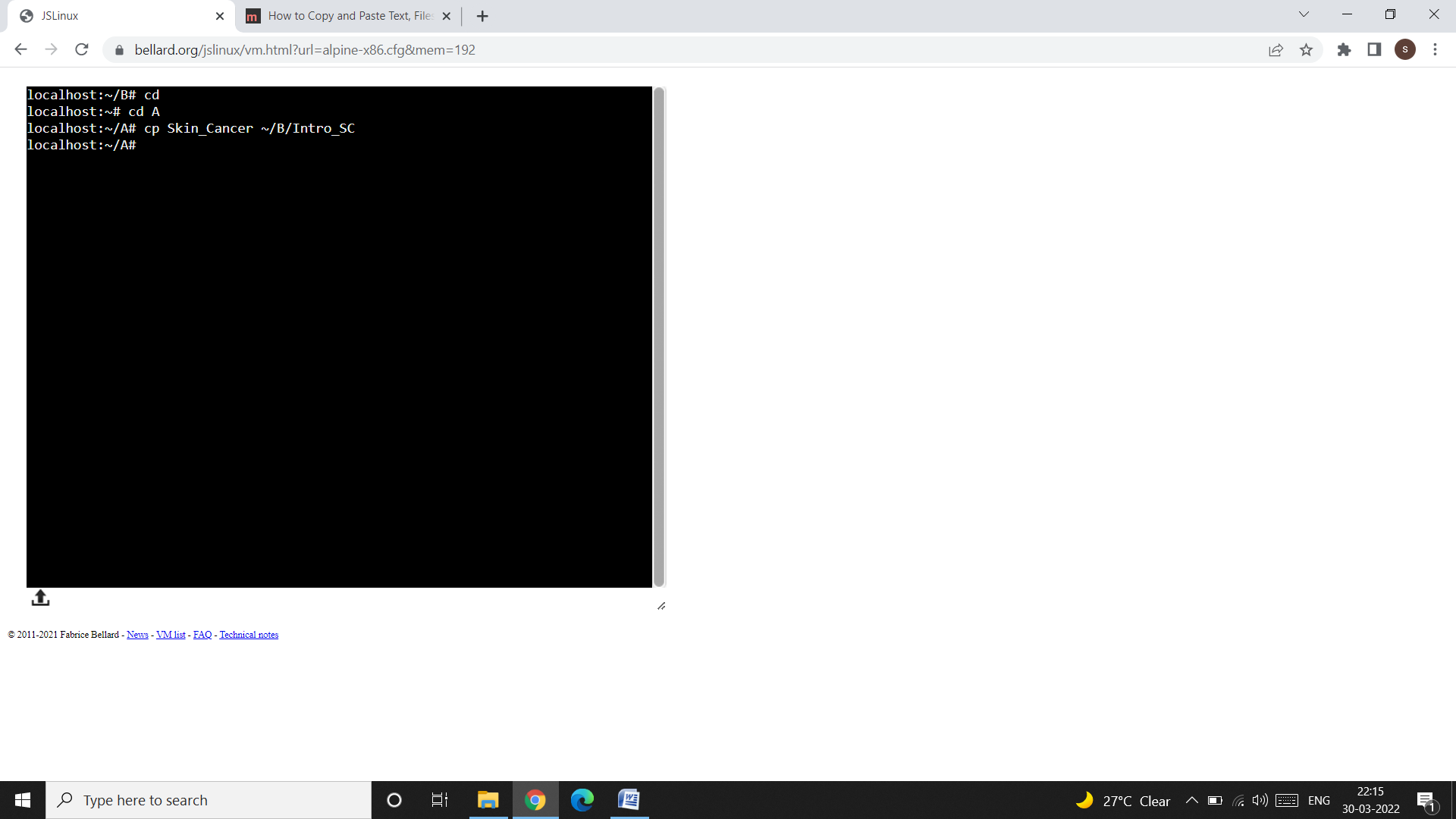


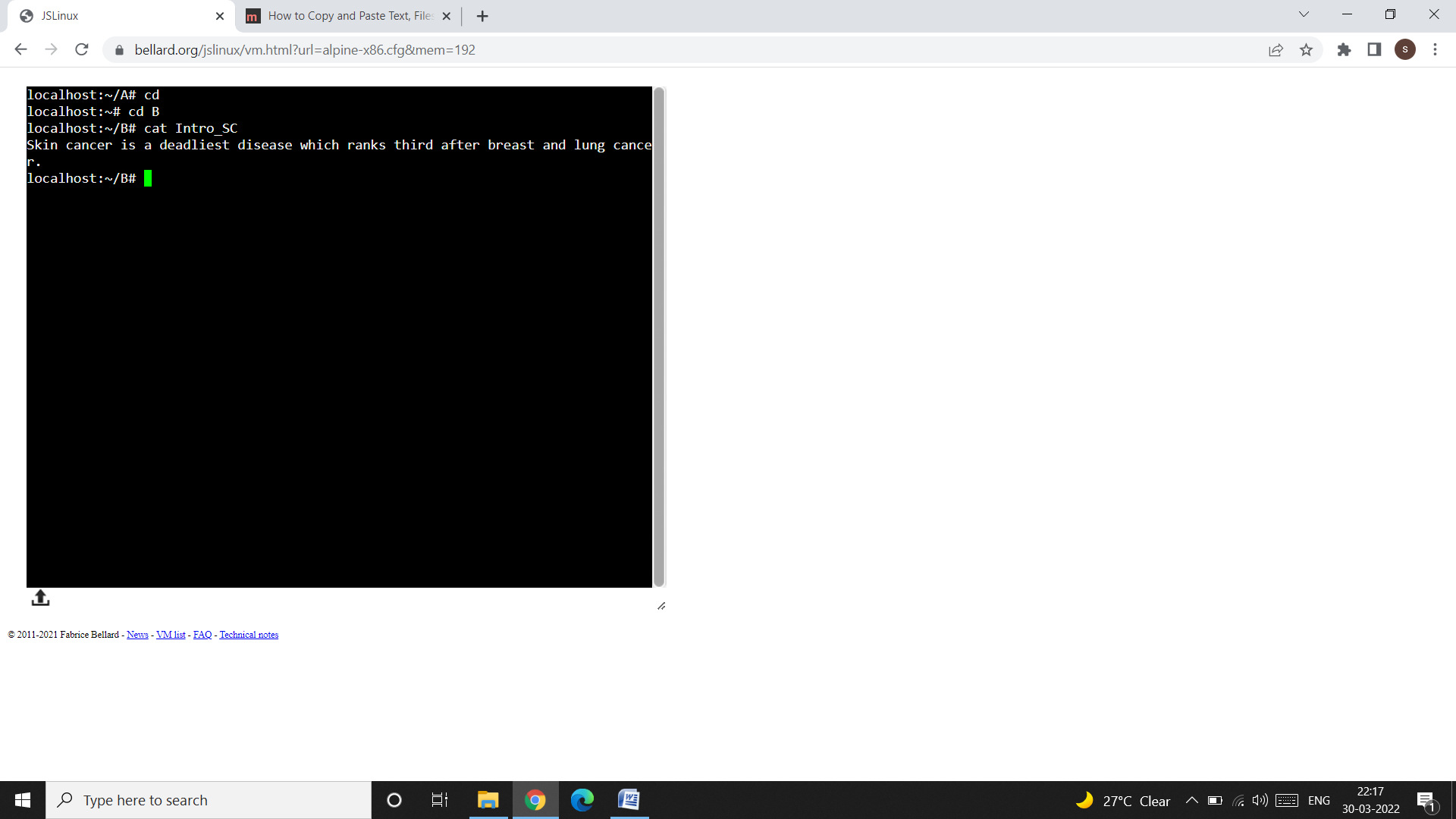
* **Note** that, if you don’t specify a filename in the destination directory (B in our case), then in the destination directory a file with the same name from which we are copying the contents will be created in the destination directory with the same contents. This is shown in the snap shot shown below:



* In the above snap shot, we can see that a file with the name Skin\_Cancer is created in the directory B. This file has the same contents as that of the file Skin\_Cancer in directory A. This is shown below:



* Now, let us copy the contents of the file Skin\_Cancer (of directory A) to the file Intro\_SC (of directory B). For this purpose, first exit from directory B and enter in directory B, like this:
* 
* Now, check the contents of the file Intro\_SC of directory B, like this:

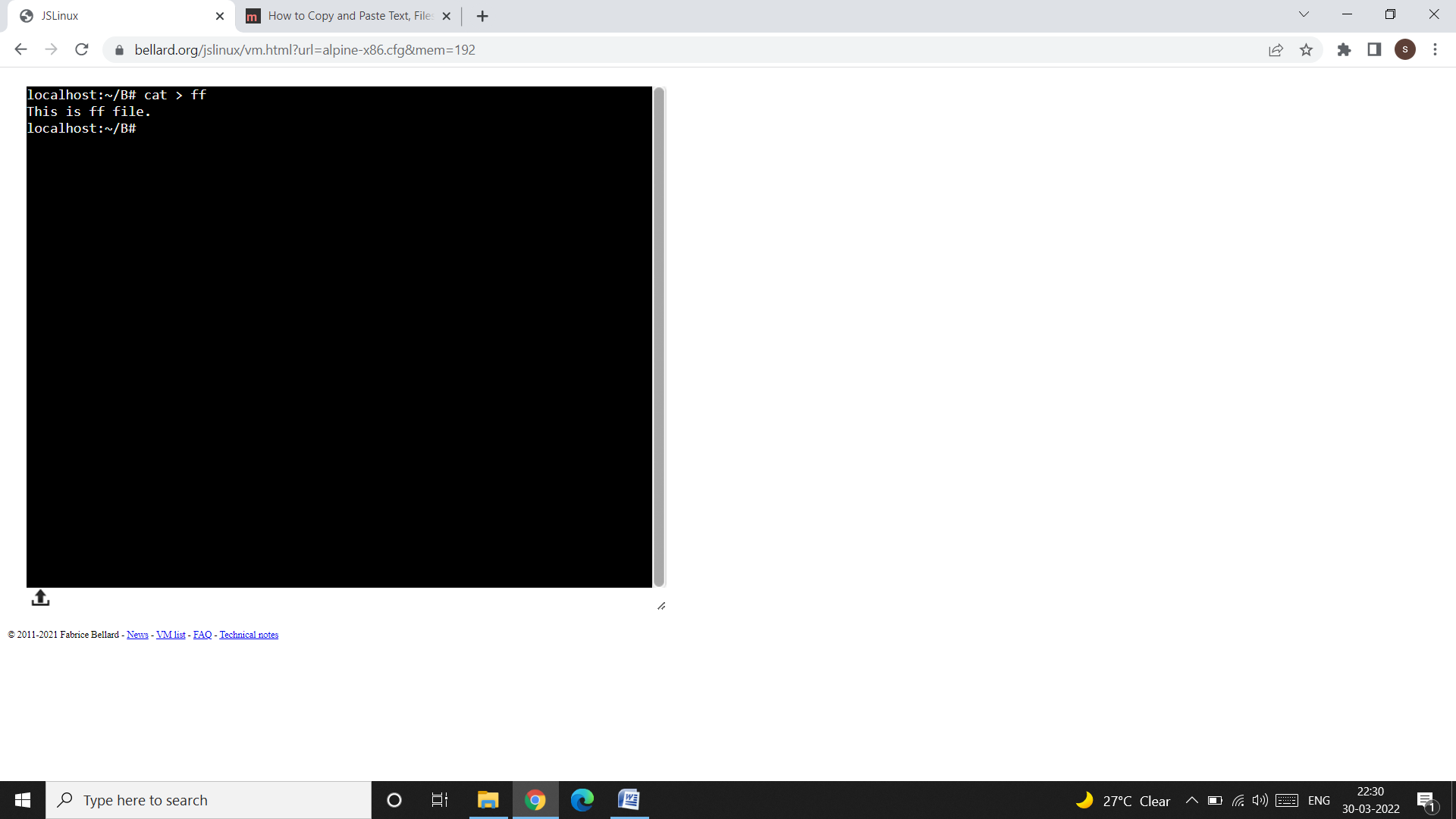


* In the above screen shot, we can see that the previous contents of the file Intro\_SC (i.e. This file provides an introduction to skin cancer.) are overwritten by the contents of the file Skin\_Cancer (of directory A).

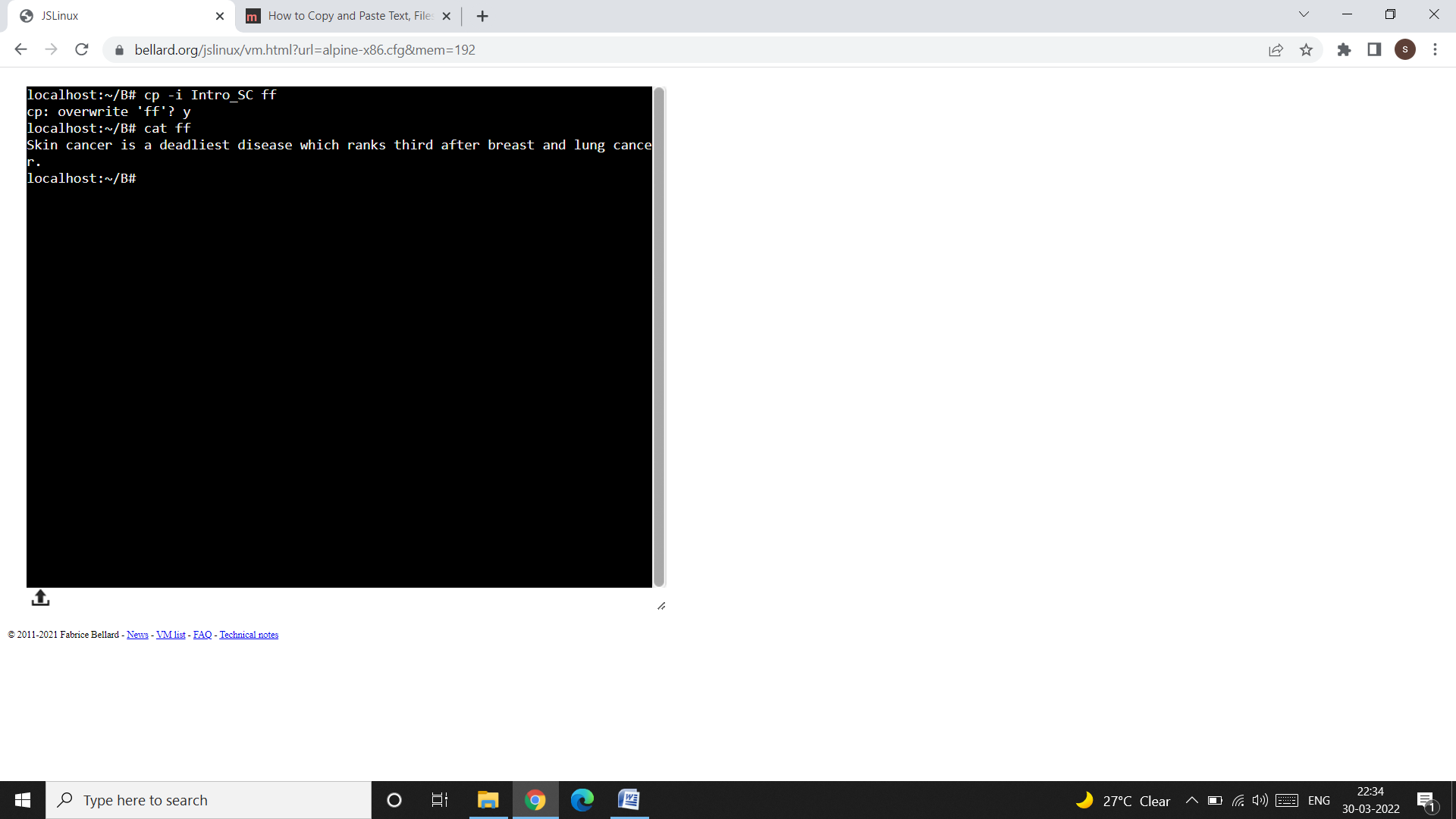
**¥cp Options:**

**Interactive Copying (-i):**

* The -i (interactive) option warns the user before overwriting the destination files.
* To understand this concept, let us create a file named as “ff” in directory B, like this:



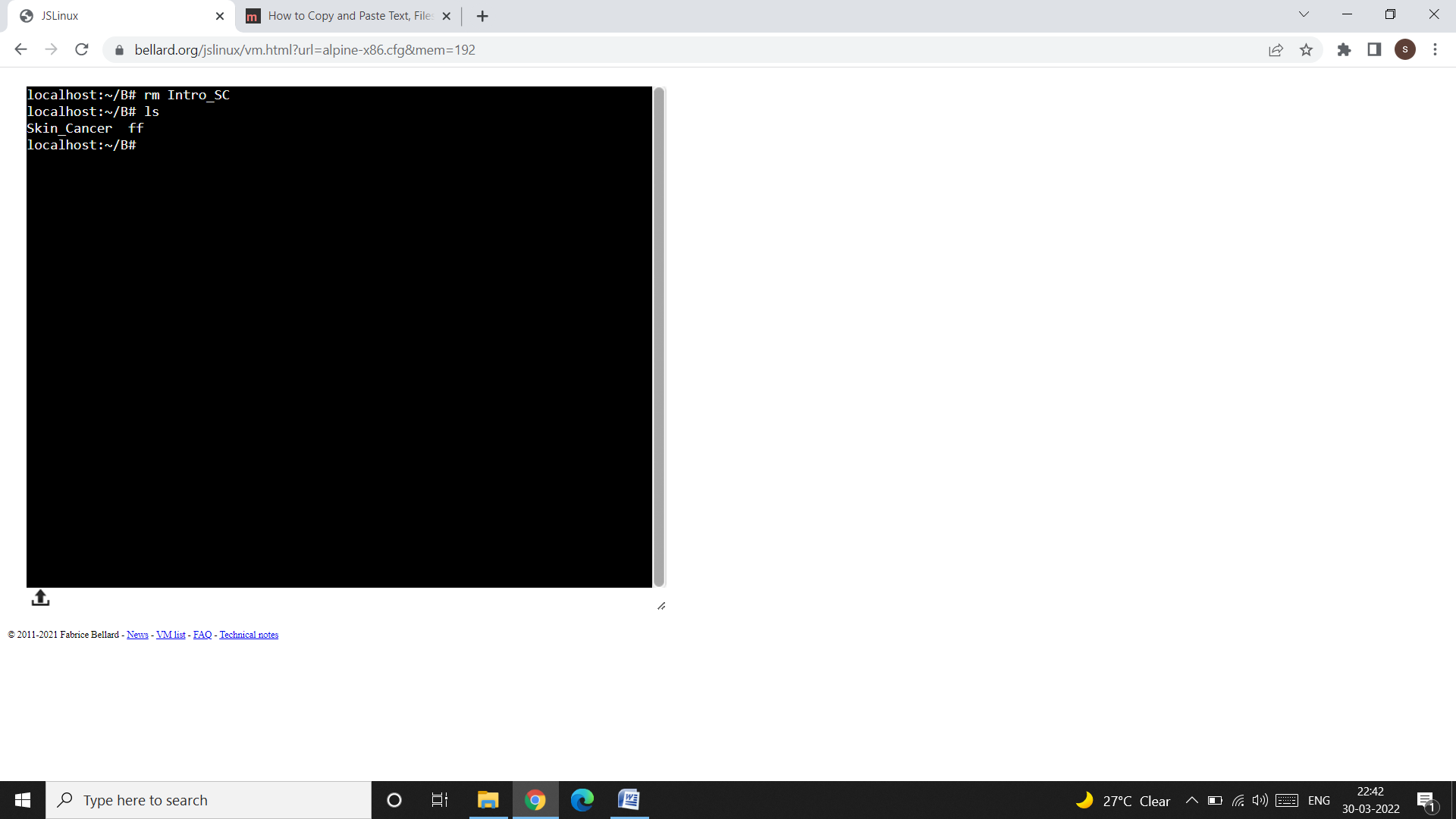
* Now, copy the contents of the file Intro\_SC (of directory B) to the file ff (of directory B) using the cp -i command like this:



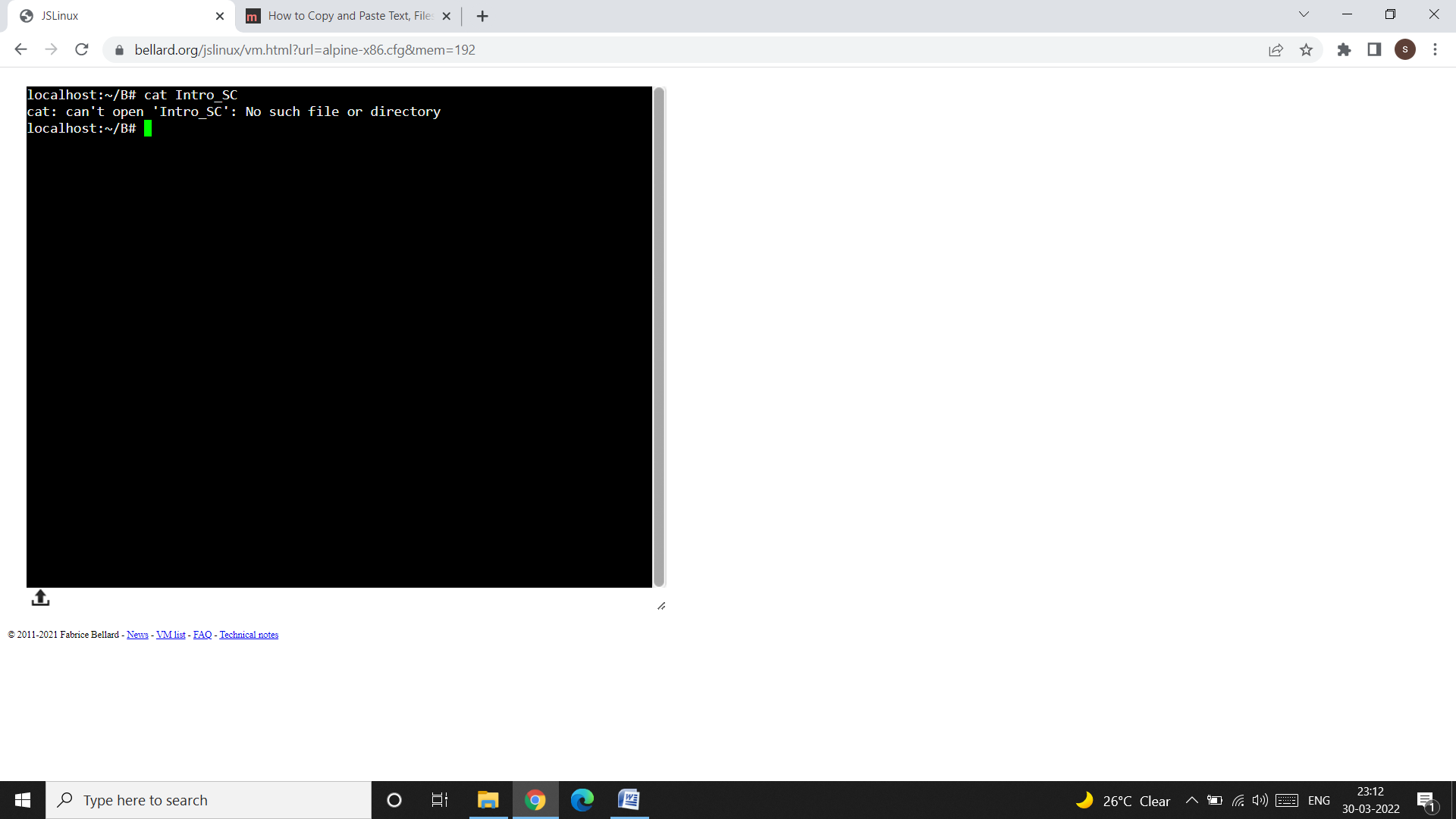
* A y at this prompt overwrites the file, any other response leaves it un copied. As I have pressed the y key, so the old contents of the file ff (i.e. This is ff file.) are overwritten by the contents of the Intro\_SC file.

**rm: Deleting Files:**

* The rm (remove) command deletes one or more files.
* A file once deleted cannot be recovered.



* In the above snap shot, we have deleted the file Intro\_SC. Any attempt to view the Intro\_SC file will give the following message:

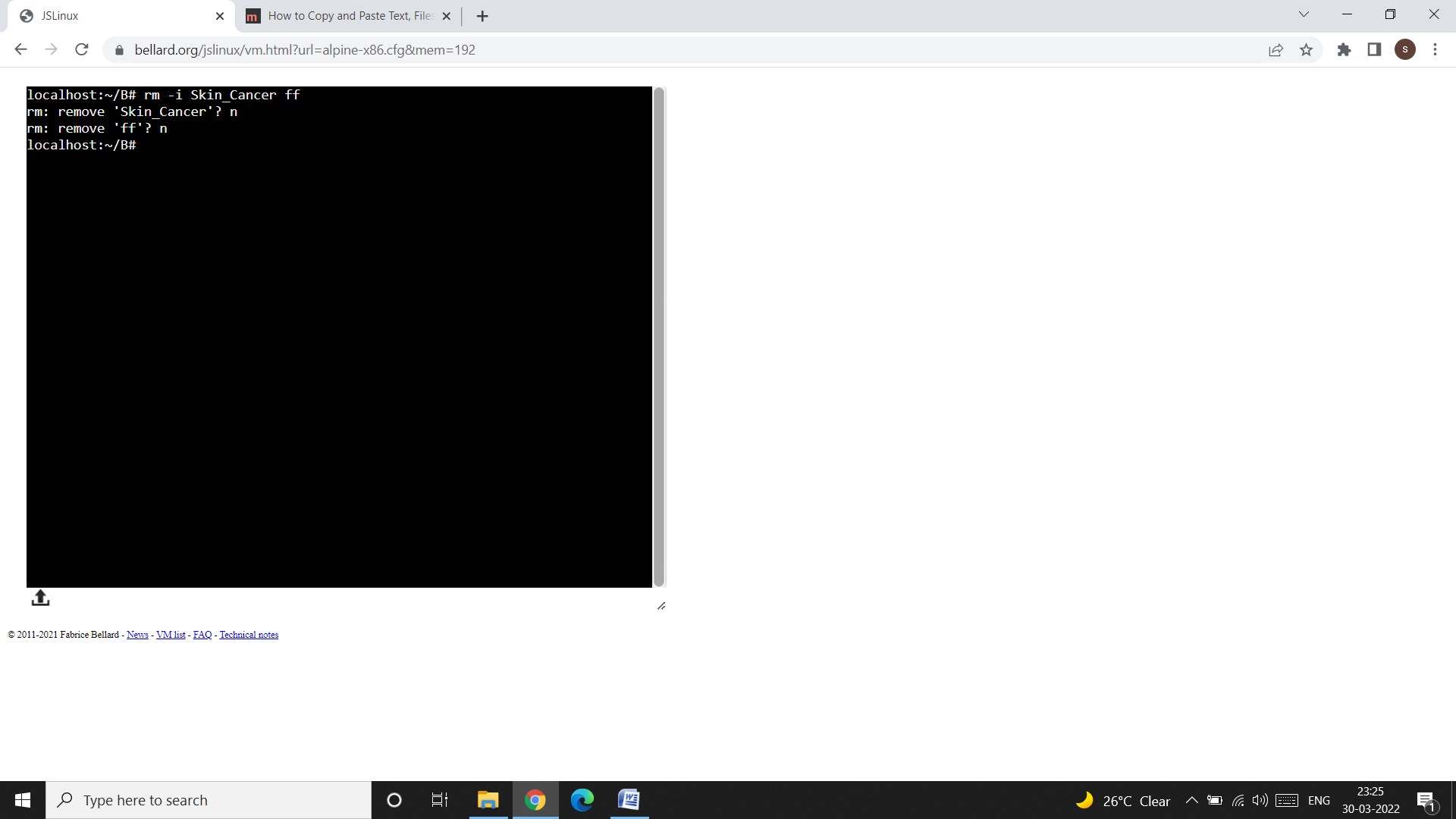


* We can delete multiple file using rm command, like this:
  + rm Skin\_Cancer ff

**rm Options:**

**Interactive Deletion (-i):**

* Like in cp, the -i (interactive) option makes the command ask the user for confirmation before removing each file, like this:

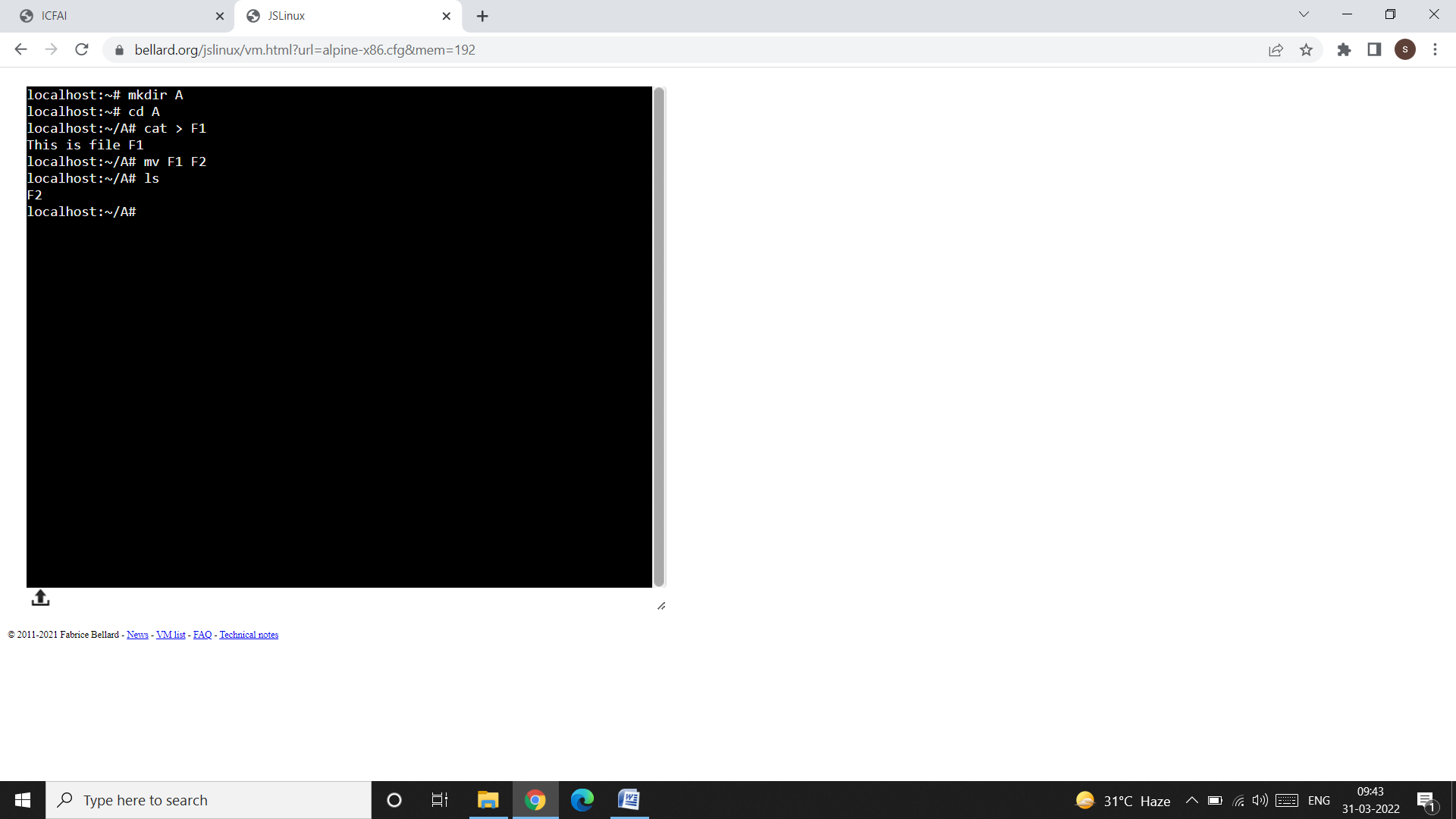


* In the above snap shot, the system first ask for deleting the files Skin\_Cancer and ff, as I have pressed the n key so the files are not deleted.

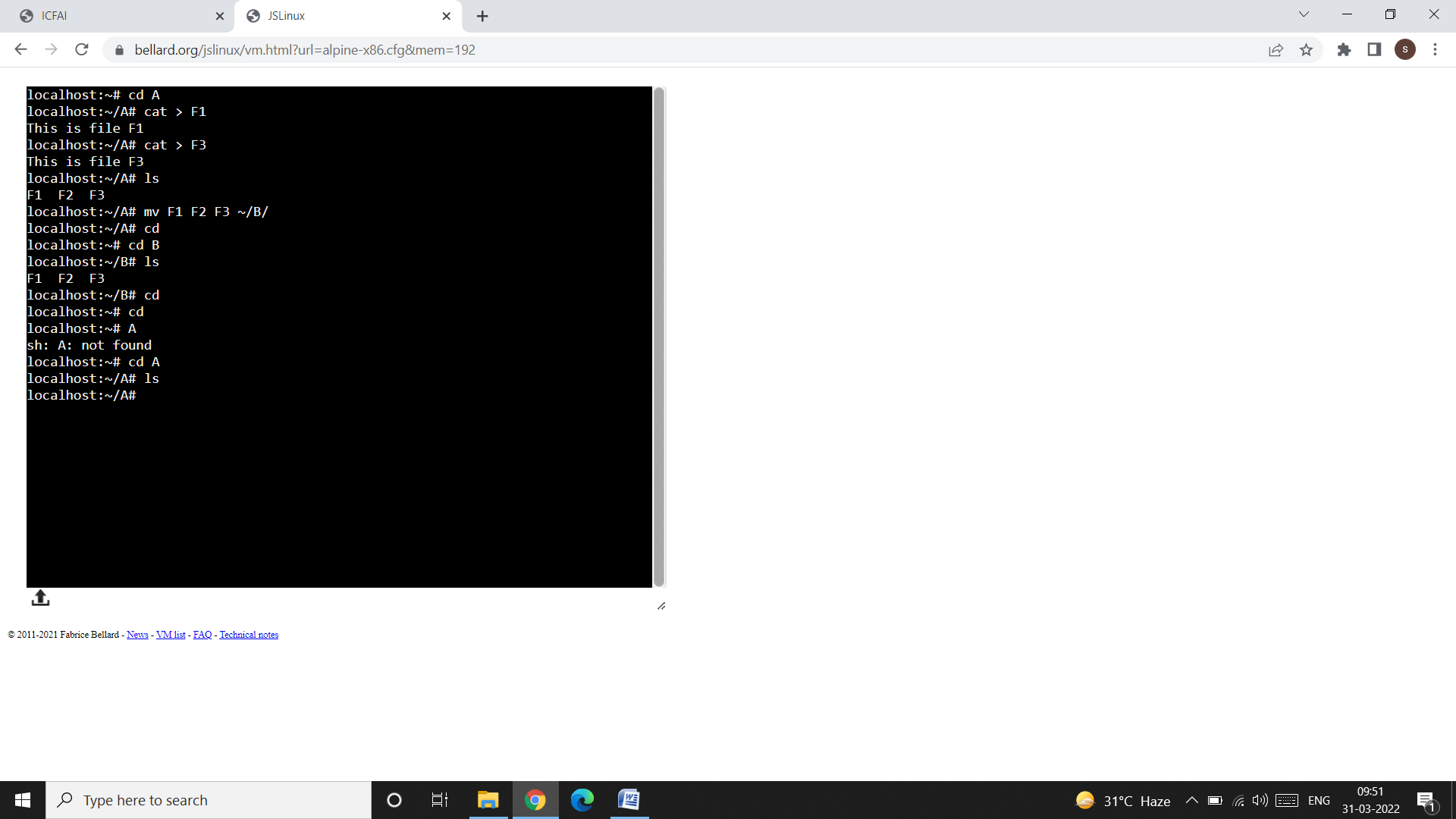
**mv (Renaming Files):**

* The mv command renames (moves) files. It has two distinct functions:
* It renames a file (or directory).
* It moves a group of files to a different directory.
* mv doesn’t create a copy of the file; it merely renames it. No additional spacebis consumed on disk during renaming.
* To rename the file chap01 to man01, you should use:

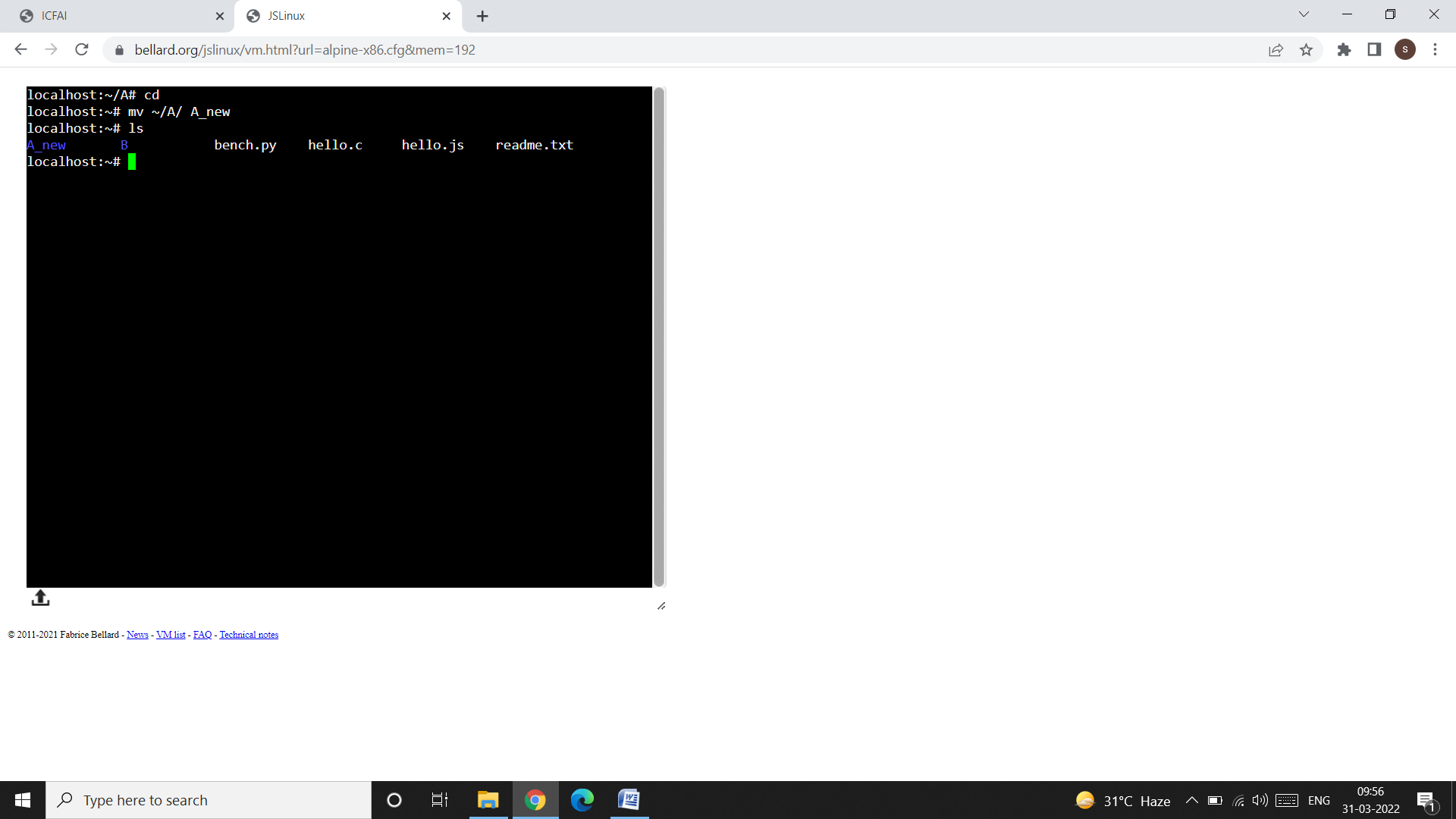
mv chap01 man01



* Like cp, a group of files can be moved to a directory.



* mv can also be used to rename a directory, like this:



* There is a -i option available with mv also, and behaves exactly like in cp.

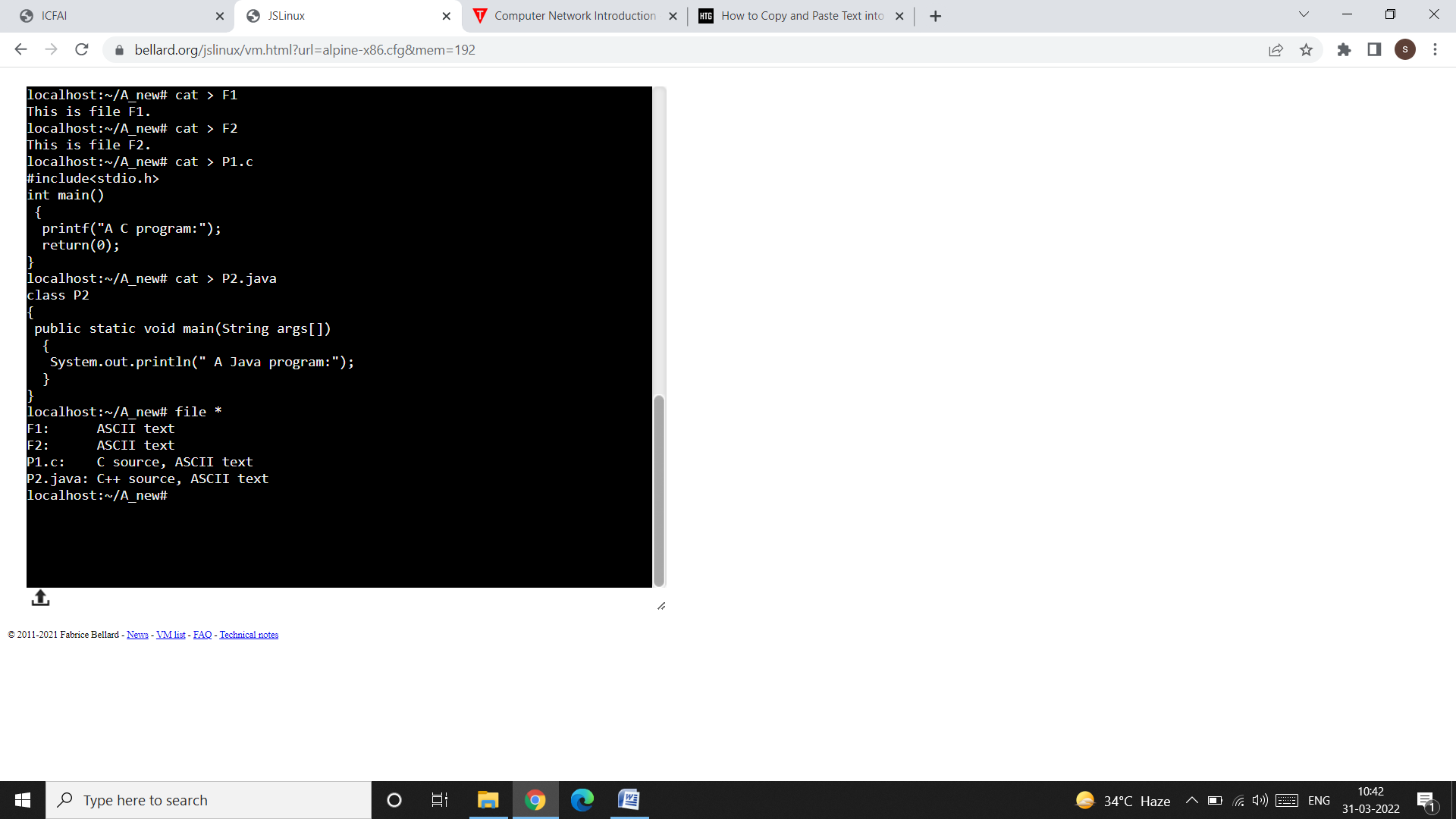
**file: (Knowing the File Types):**

* UNIX provides the file command to determine the type of file, especially of an ordinary file. The syntax of this command is:

file filename

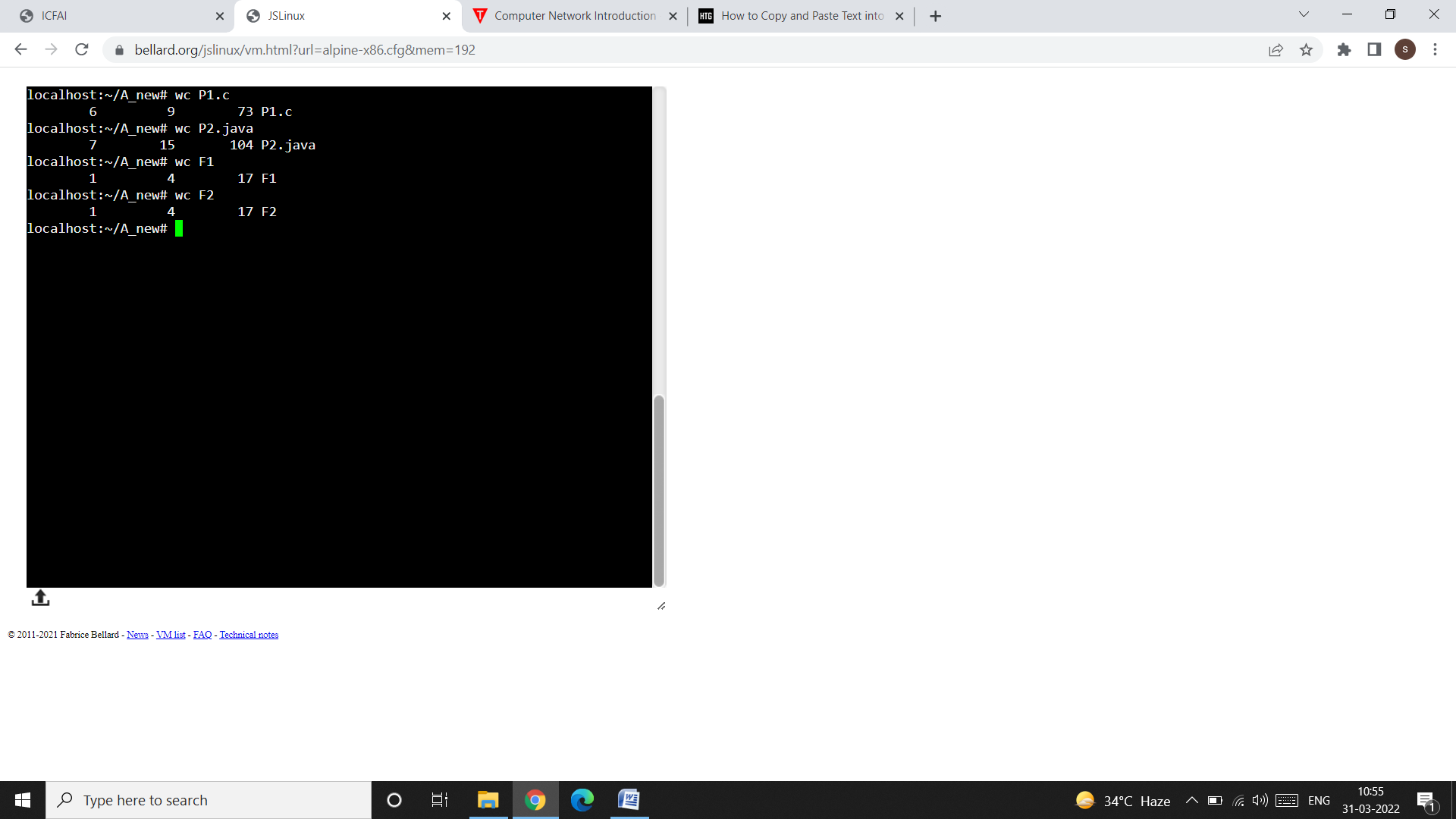
Example: file xyz

* We can use \* to signify all files in a directory.

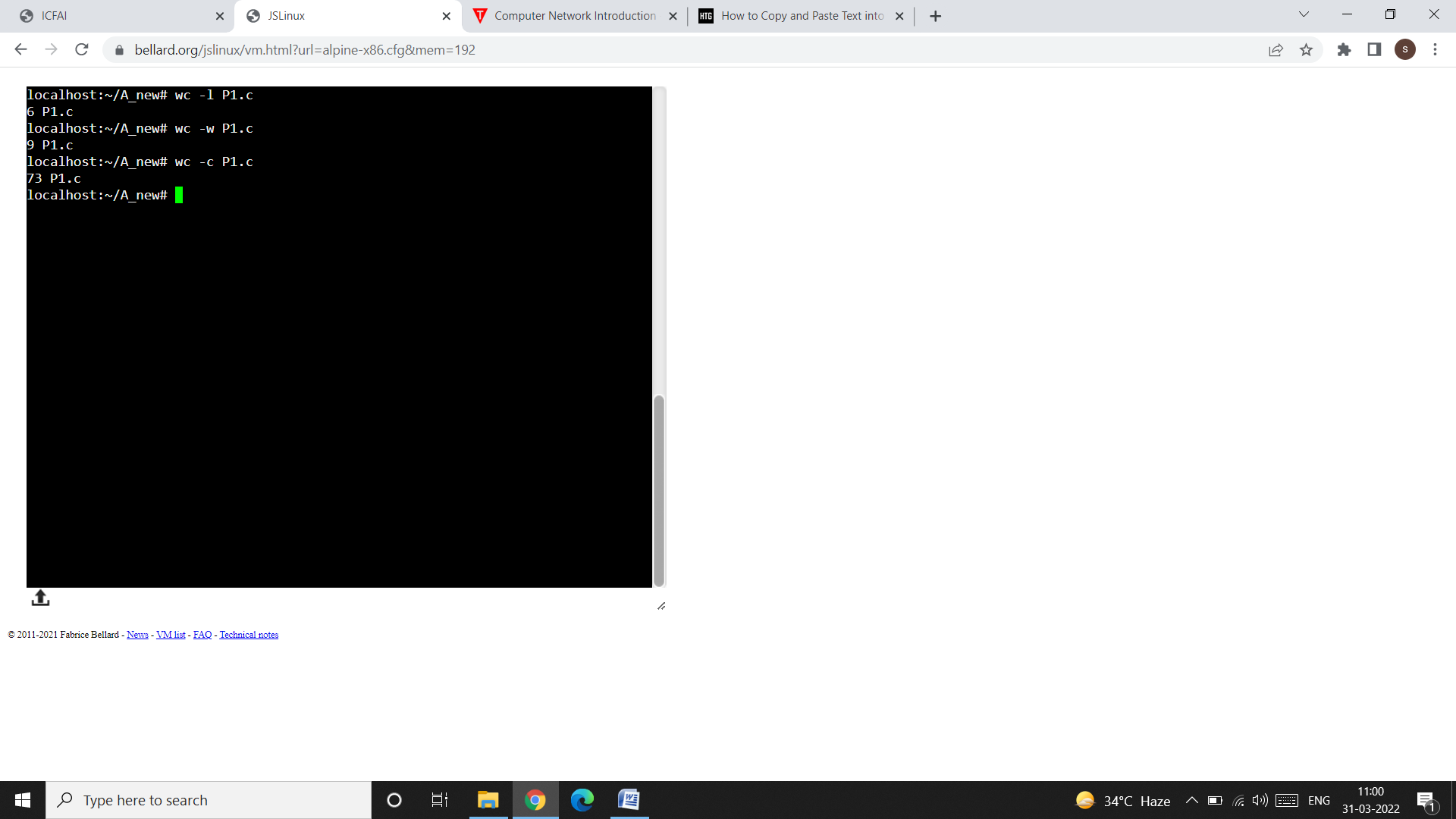


**wc: Counting Lines, Words, and Characters:**

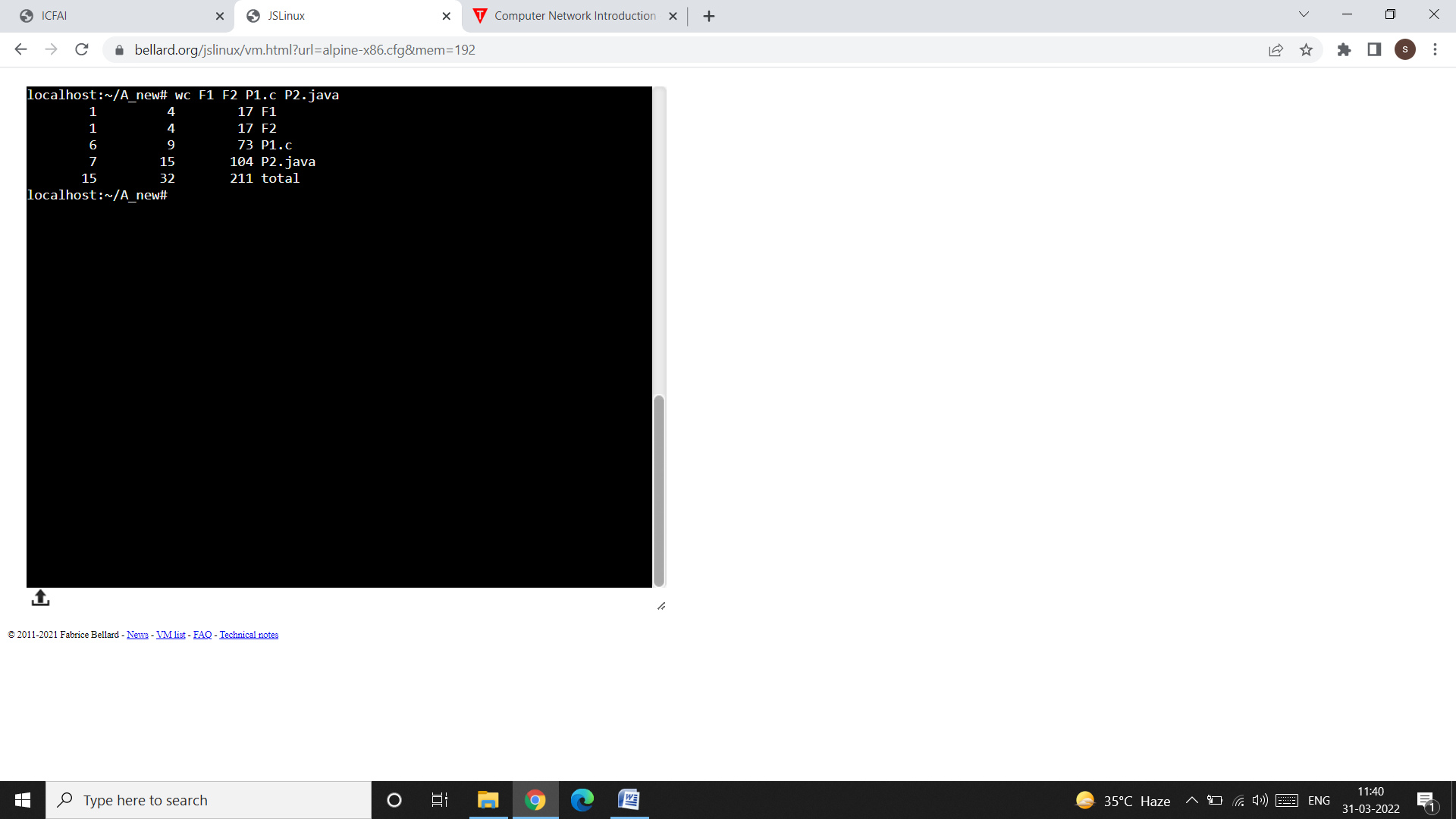
* wc is a word-counting command that also counts lines and characters.
* It takes one or more filenames as arguments and displays a four-column output.



* wc offers three options to make a specific count. The -l option counts only the number of lines, while the -w and -c options count words and characters, respectively.

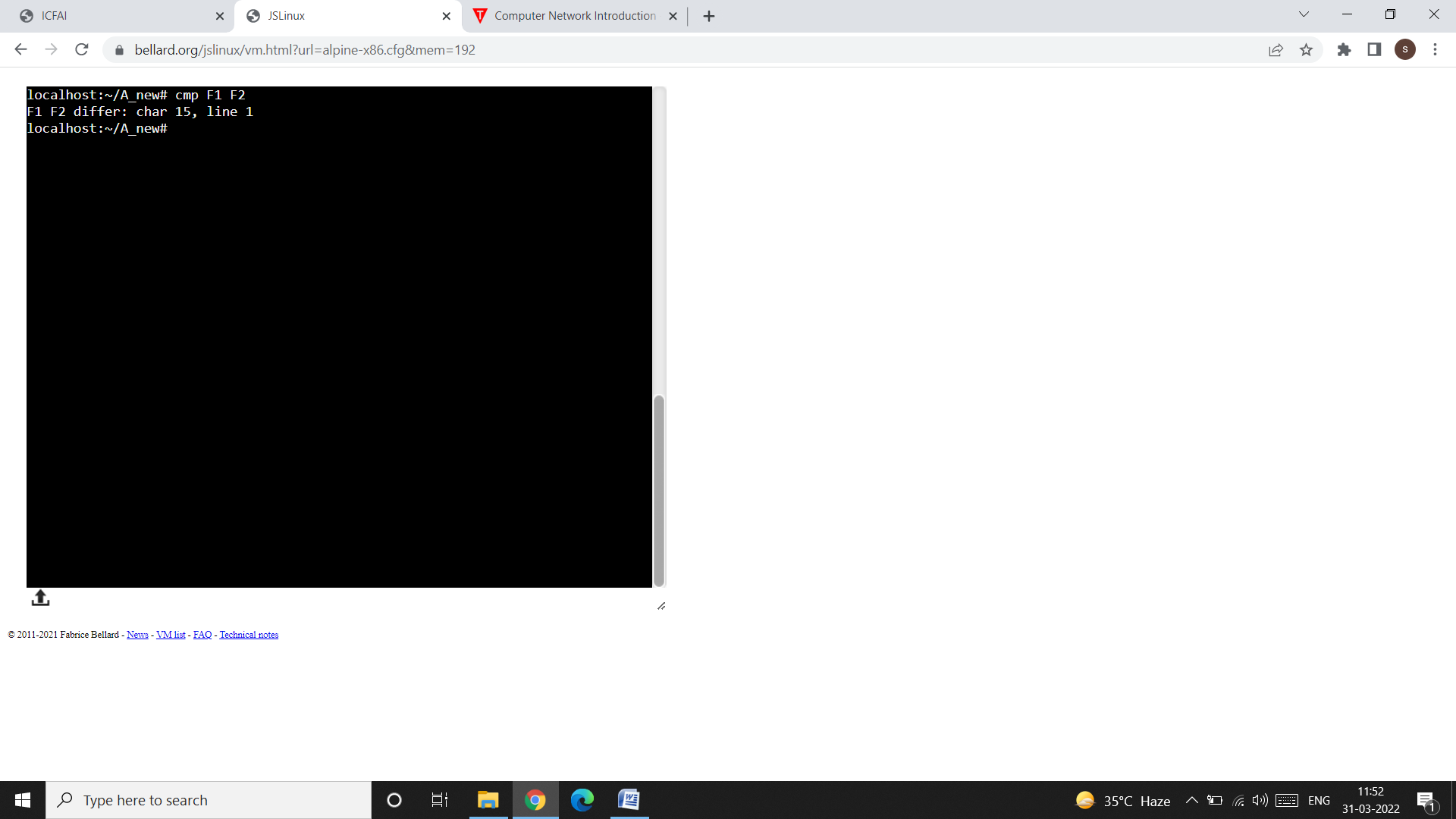


* When used with multiple filenames, wc produces a line for each file, as well as a total count:



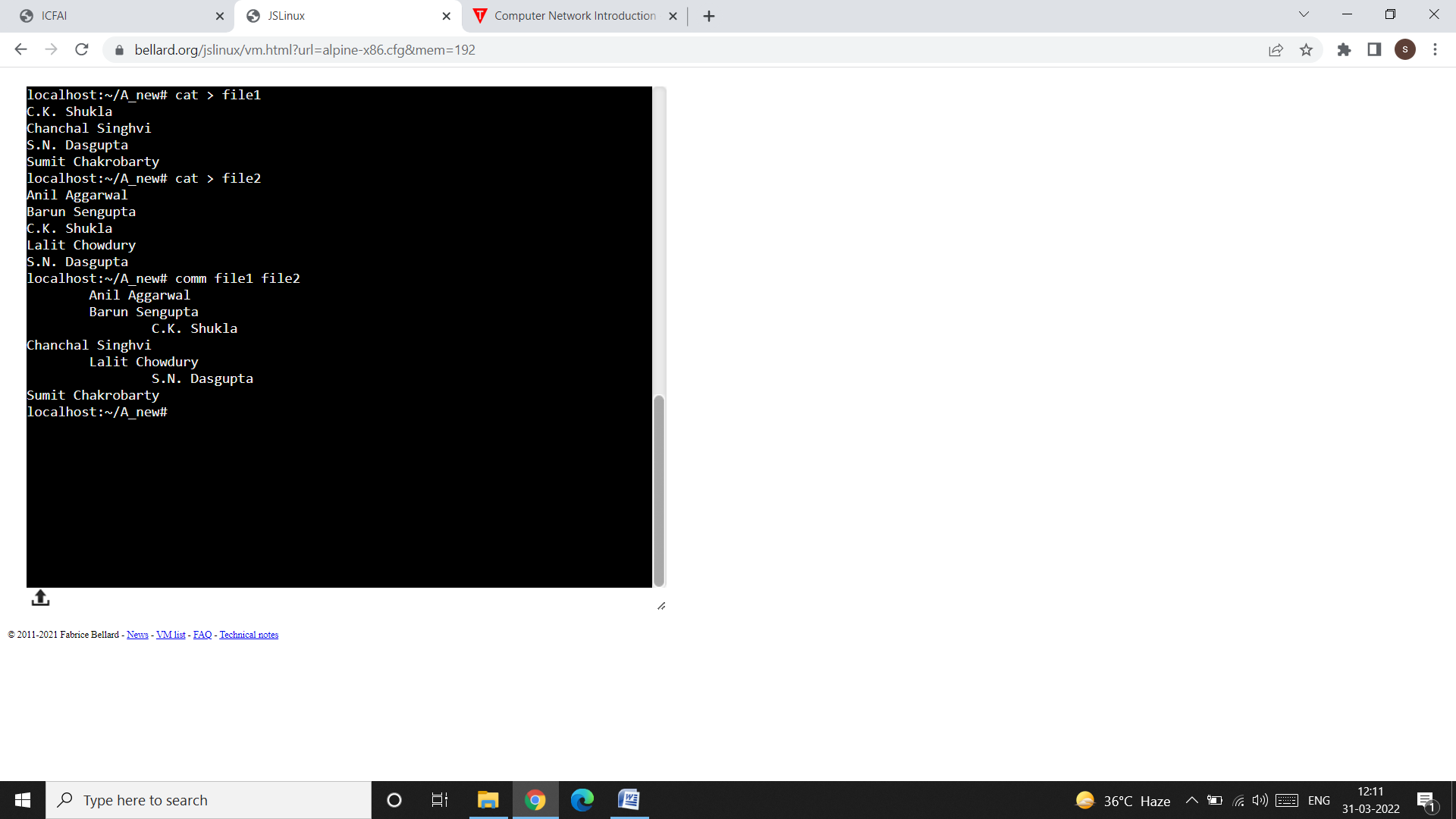
**cmp (Comparing two files):**

* The cmp command is used to compare two files.
* This command requires two filenames as an argument.
* The two files are compared byte by byte, and the location of the first mismatch is echoed to the screen.
* By default, cmp doesn’t bother about possible subsequent mismatches but displays a detailed list when used with the -l (list) option.
* If two files are identical, cmp displays no message, but simply returns the prompt.



**comm (What is Common);**

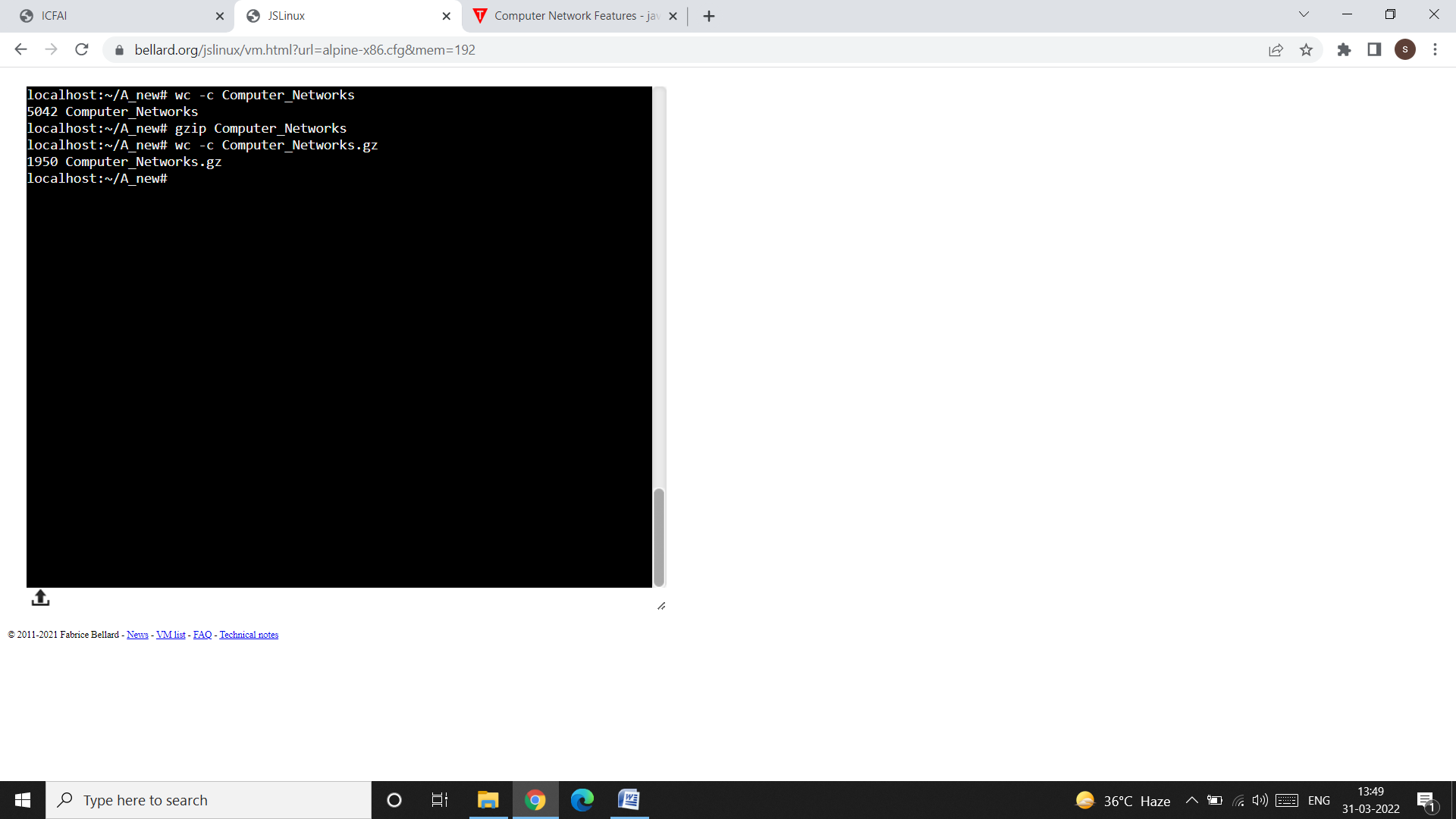
* Suppose you have two lists of people and you are asked to find out the names available in one and not in the other; or even those common to both. comm is the command you need for this work.
* It requires two stored files, and lists the different entries in different columns.



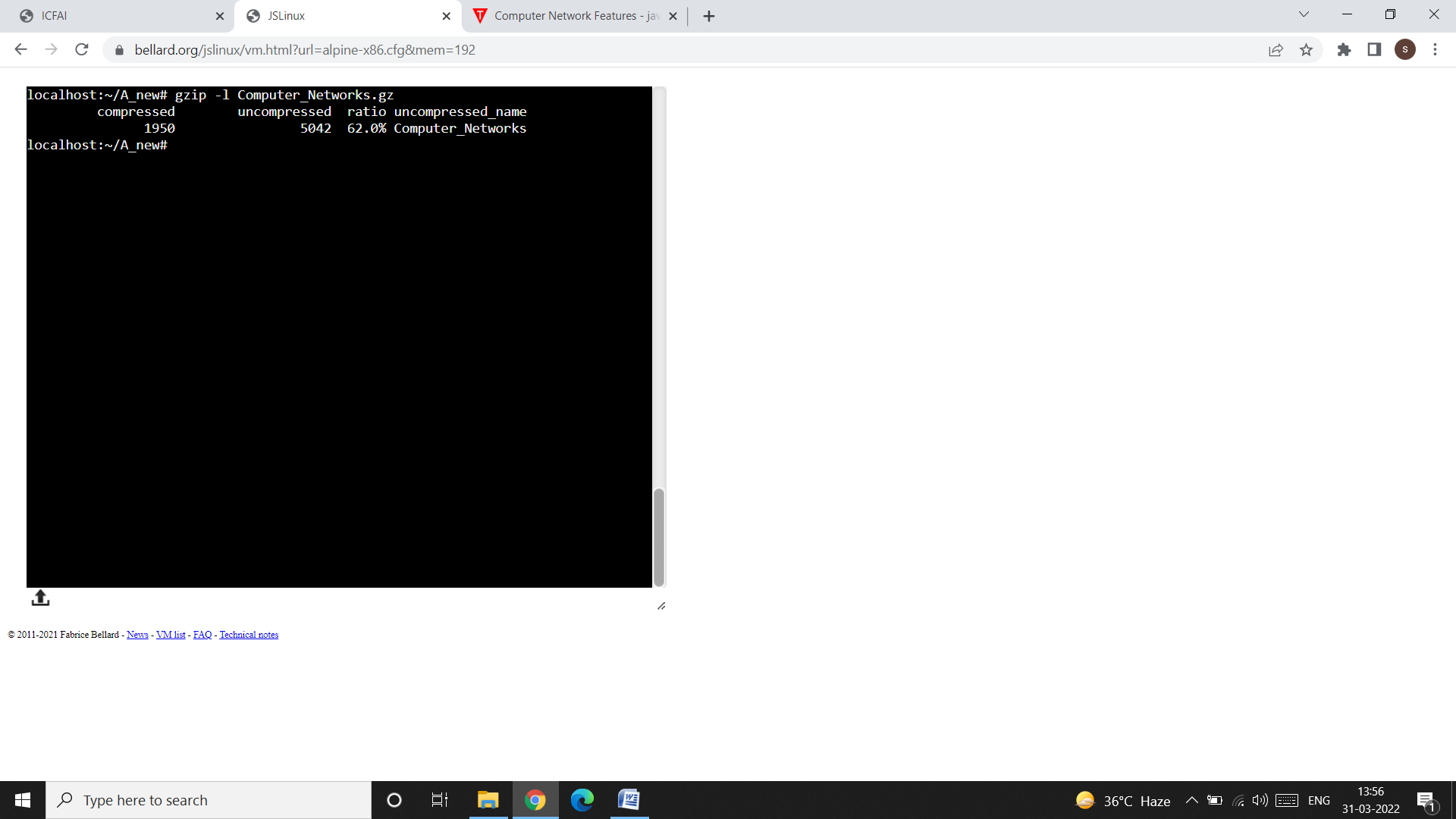
* Both files are stored and have some differences. When you run comm, it displays a three-column output.
* The first column contains two lines unique to the first file, and the second column shows three lines unique to the second file. The third column displays two lines common (hence its name) to both files.

**gzip and gunzip: (Compressing and Decompressing files):**

* The gzip command is used to compress (to reduce the size) a file.



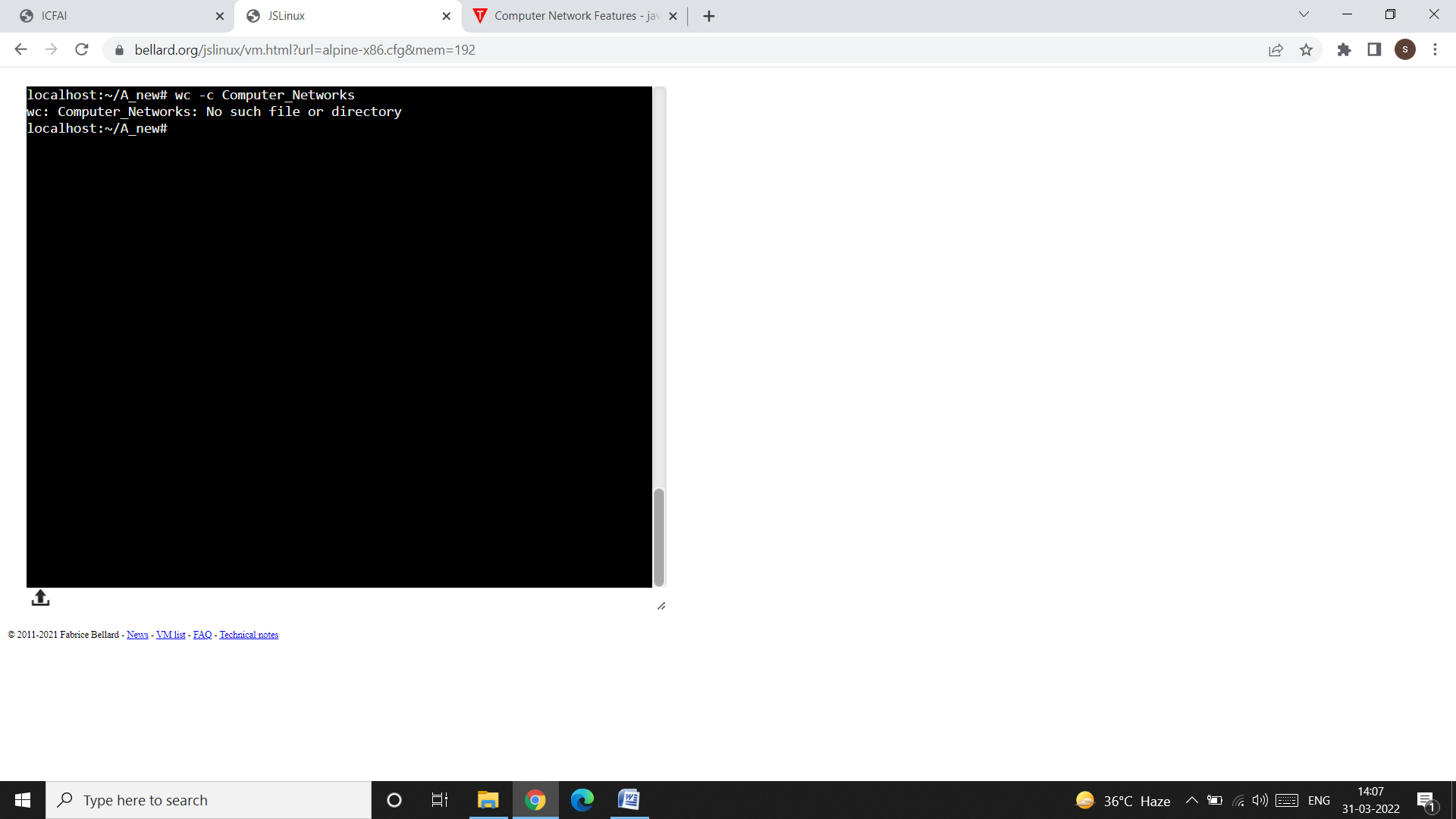
* We can use -l option with the compressed or original filenames as an argument; for knowing how much compression we actually achieved.



**gzip Options:**

**Uncompressing a “gzipped” File (-d):**

* To restore the original and uncompressed file, we have two options: Use either gzip -d or gunzip with one or more filenames as arguments; the .gz extension is optional yet again.
* Remember that, if you try to access the file Computer\_Networks before decompression (or uncompression), you will get the following message:



This is because after compression the Computer\_Networks file is changed to Computer\_Networks.gz. After decompression, the file Computer\_Networks.gz will again be changed back to Computer\_Networks.

Compression

Computer\_Networks Computer\_Networks.gz

Computer\_Networks.gz Computer\_Networks

Decompression

