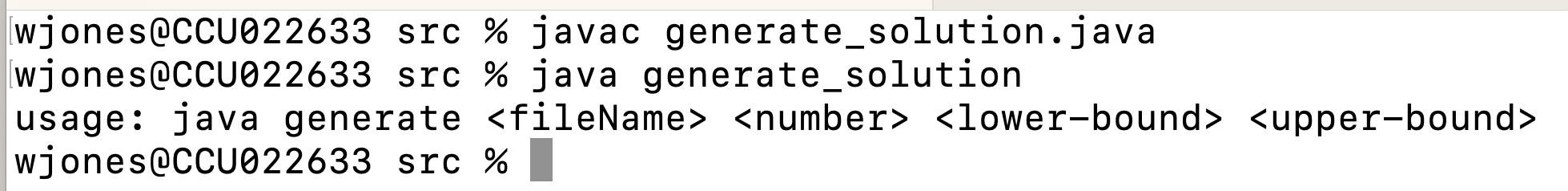
Here you should start with the code we had in class, [as seen in this video](https://coastal.yuja.com/V/Video?v=9489703&node=41664451&a=125779960&autoplay=1). Then modify it to have the following interface:



And this behavior:

A screenshot of a computer program

Description automatically generated

See, the program wrote 5 random numbers into the specified file. The numbers were distributed *uniformly* UNIF[lower,upper], inclusive, where upper and lower were parameters specified at the commandline.

The resulting file is shown with the included 5 integers. **The program also shows the elapsed runtime (in seconds).**

If the program is run repeatedly, we get different numbers in appearing in the file:

A screenshot of a computer program

Description automatically generated

We can see that as we increase the number of integers to be written into the file, the total elapsed time increases:

A close-up of a computer code

Description automatically generated

We can also see at the resulting file still contains the correct number of lines (this uses the word-count (wc) utility program – you can do the same thing in Windows ([ChatGPT link](https://chat.openai.com/share/fc689a1b-6dee-4ca0-a758-432c6921e204)).

A screenshot of a computer

Description automatically generated

Note the file size. Note the use of the Linux ‘**tail’** command to see the last few lines in the text file. (**here is** [**how you do it in Windows**](https://chat.openai.com/share/427b3d39-37ea-449d-bc61-b9f89a093c66))

TODO:

1. Complete your program so that it has this **EXACT SAME BEHAVIOR**
   1. Same usage statement
   2. Same print statements (exactly)
   3. Same formatting to screen
   4. Same functionality
   5. Make sure you handle the exceptions with try-catches.
2. Submit two things
   1. Your program, named generate\_solution.java (**NOT THE .CLASS FILE**)
      1. Make sure the code is well documented
   2. A MS Word document (NOT PDF) that has a full report of what you did, how you solved the problem, incorporate screenshots from the code
      1. Show screenshots of you
         1. Compiling your code at commandline
         2. **Running your code at the commandline with at least the same parameters as I use above, but better yet, others that show something interesting**
         3. When you do that – make sure I can see you can do cat / type, and other various windows / linux commands
3. **SUBMIT ALL THAT IN MOODLE – i.e. TWO ITEMS (the code, and the report)**
   1. **After submitting to Moodle – on a separate tab, go back into the assignment and download your submitted work back into a temp folder in your Downloads folder to make sure that what you submitted is actually there and is what you intended to submit.**

**Do not submit late – late work is not accepted.**

WMJ