ls command lists files under the provided or, if none is specified, current directory. The “-l” option is used to display the files as a list rather than in a grid and with many details such as permissions, file size, creation date and time, and the owner user and group. I picked this command, because it is an extremely useful tool that I have used countless times during my internship this summer to view the files I dealt with. One simply cannot do without it if they are working on a remote Linux server.

Process hierarchy of my code:

* The shell prints the first line.
* The shell spawns a child process by fork() (hereafter referred to as *parent*).
  + *parent* prints the second line.
  + *parent* spawns a child process by fork() (hereafter referred to as *child*).
    - *child* prints the third line after *parent* terminates (it waits for output from *parent* to complete, that is, for reading from the pipe to complete).
    - *child* arranges it so that the output is written to “output.txt”, executes “grep -e “-l” -A 2” on *parent*’s redirected output, and terminates.
      * Note: *parent*’s output is dynamically processed by *child*, as *parent* sends man’s output directly to *child* as it is being processed – they work concurrently.
  + *parent* executes “man ls” and terminates.
    - Note: *parent* sends man’s output immediately to *child*, the complete output is transmitted in pieces.
* The shell prints the last line after waiting for all child processes to terminate.

To use the coding given in the guide document, my program is 1a. man and grep are in a parent-child relationship, because the process that runs man is also the process that spawns the process that runs grep by fork() - the shell does not spawn two children, but rather a child and [indirectly] a grandchild. They can run concurrently, because the process that runs grep does not explicitly wait for its parent to finish. Instead, the processing order is solely dependent on communication via the pipe – while reading the pipe implicitly blocks the parent process (man) from proceeding, it happens every time man sends a page to grep. Therefore, the child process (grep) runs every time an output is sent by the parent (man).