Operating Systems	Lab Final Exam

Name:

(0.5 points) 1. What can we do with a "pipe" in the command line? Which character is associated with the pipe in the shell? Give an example of usage.

(0.5 points) 2. The environment variable PATH is quite important in UNIX. Explain what its purpose is and how we can check its value.

(0.5 points) 3. Create a file named "myfile" and change its permissions so just you can access its contents. Write down the commands you used.

(0.5 points) 4. Which signal would you send to a process so that it ends in an orderly manner? If the process does not end, which other signal do you use to guarantee its finalization?

(0.5 points) 5. Which command do we usually use to watch the processes running in the system? Write down an example of use that shows, at a given moment, an extended list of ALL processes running in the system.

(0.5 points) 6. Given the program “reserva”, execute it, and before it ends, check the memory usage of the process (the program is called without arguments: ./reserve).
How much memory does the process occupy? How much memory does its stack occupy? Write down the steps followed to obtain the answers.

(0.5 points) 7. Which C function do we use to create a new process and what are its returning values? Which code does the new process execute when it runs for the first time?

(0.5 points) 8. Write down the process identifier (PID) of the Shell in which you are doing the exam, and the id of its parent process. Write down the steps followed to obtain these values.

(6 points) **Exercise**

You have got 10 files with N+1 integers each (N being different in each file). The first integer in each file is the number of integers that follows (N). That is, if the first integer is 120, there are 120 integers stored in the file (plus the first one). The names of the files are "TXX.txt", where XX is a number from 01 to 10 (i.e. T01.txt, T02.txt…T10.txt).

Design and develop a program in C language with the following description:

- Main process must create a results file named "results.txt" where each thread should write.
- Main process must create two threads (th_A and th_B) that do the following tasks:
    - th_A reads each file (TXX.txt) in ascending order (from T01.txt to T10.txt) while th_B reads each file in descending order (from 10 to 01).
    - Each thread must load the contents of the file it is working on in a dynamically allocated array (in the heap). The number of integers in the corresponding file determines the size of the array, and the array should be free as soon as the thread finishes working with it (before opening a new file).
    - th_A must obtain the minimum value in each file, and write it down in the results file with the following format:
      "`File TXX.txt: min=Y`", where "Y" is the value of the minimum found.
    - th_B must obtain the maximum value in each file, and write it down in the results file with the following format:
      "`File TXX.txt: max=Y`", where "Y" is the value of the maximum found.
    - Files must be written avoiding conflicts. Both threads must be able to read a file simultaneously, but only one at a time can write in a given file.
- When both threads have finished their jobs, the main thread must close the results file.