Text of exercise #2

Matteo Corain S256654

Laboratory #10 – System and device programming – A.Y. 2018-19

A C program is run with N parameters. Each parameter indicates a relative or an absolute path to a file system directory tree. The program has to compare the content of all directory trees to decide whether they have the same content or not.

Two directory trees have the same content if and only if all directory entries (files and sub-directories) have the same name (excluding the path leading to the root, which differ but for self-comparisons). Upon termination the program has to state whether all directory trees have the same content or not.

Suggestions:

- The main program run one “reading” thread for each directory tree plus one unique “comparing” thread.
- Each “reading” thread visits one of the directory tree. It is reasonable to supposed (even if this is not explicitly stated by the system call specifications) that in case of equivalent directory trees, all visits proceed using the same order, i.e., they deliver all directory entries in the same order.
- Reading threads synchronize themselves for each entry they find, waiting for each other before moving to the next entry.
- For each entry, the “reading” threads activate the “comparing” thread.
- The “comparing” thread compares the name of all entries received. It stops all other threads (and the program) in case the entries are not equal. Otherwise, it returns control to the “reading” threads.

Notice that there are at least 3 possible termination conditions to manage:

- Directories are indeed equivalent. This should lead to a successful termination.
- Directories differ for some specific entry name. This can be intercepted by the comparing thread.
- Directories are (partially) equivalent but they include a different number of entries. In this case, one thread may terminate its reading task before all others which may be waiting for it forever. This situation should be avoided.