Makefile:
[4] Creare un makefile che compili il programma solamente se il file passato da terminale `FILE=<path/to/File.txt>` esiste. Se non esiste deve essere restituito un messaggio di errore. Qualora il file dovesse esistere, il makefile deve creare una nuova cartella (nella directory in cui viene eseguito) e copiarvi dentro il file. Dopodichè, dovrà compilare i sorgenti creando un eseguibile sempre all'interno della nuova cartella. Il nome dell'eseguibile e della cartella sono dati da terminale con altre 2 variabili: `DIR=<path> EXE=<name>`.

Programma in C
Si deve creare un programma in C che riceva come unico argomento il percorso completo (con il nome) di un file (per esempio `/path/to/file.txt`). Il file, che in fase di valutazione sarà già esistente, conterrà diverse righe (separate dal carattere a capo, '`\n`') che potranno avere la seguente sintassi (ogni argomento non contiene spazi):

- `kill <signo> <pid>`
- `queue <category> <word>`
- `fifo <name> <word>`

Il programma dovrà fare un controllo sui parametri [2], gestendo eventuali errori con messaggi su `stderr`. Inoltre, il programma dovrà leggere il file ed interpretarne il contenuto con l'esecuzione delle seguenti funzionalità:

- [4] **`kill <signo> <pid>`**: il programma manda un segnale **`<signo>`** al processo identificato da **`<pid>`** (NB: solo quel processo dovrà ricevere il segnale!).
- [5] **`queue <category> <word>`**: il programma invia sulla coda associata alla tupla (**`/path/to/file.txt,1`**) un messaggio di tipo **`<category>`** contenente la parola **`<word>.`** Solo qualora la coda non esistesse, essa deve essere creata.
- [5] **`fifo <name> <word>`**: il programma invia un messaggio sulla fifo **`<name>`** contenente **`<word>`**. Solo qualora la fifo non esistesse, essa deve essere creata.
- [7] Il programma deve usare threads separati per gestire i vari comandi (un thread per ogni categoria di comando), ed usare il thread principale solo per la lettura del file.
- [3] Il programma riceve un acknowledgment con un signal `SIGUSR1` dopo l'esecuzione di ogni comando, e solo dopo questa ricezione può eseguire il comando successivo. **NB**: non è richiesta l'implementazione dell'invio dell'acknowledgment. Saremo noi in fase di valutazione ad inviare il SIGUSR1.

Eventuali attese di 1 secondo sono tollerate nella gestione dei comandi. Il programma deve essere in grado di operare per un numero infinito di comandi. Dopo averli interpretati tutti deve terminare. Tutti i parametri hanno una lunghezza massima di 255 bytes.
NB: la compilazione dell'eseguibile non deve generare alcun messaggio di warning da parte di GCC! Non usare nessun flag diverso da -pthread.

Bash:
[4] Scrivere uno script bash, con shebang, che accetta 10 parole (gestire con un errore i casi in cui ne vengano date meno o più) e le restituisce su standard output in ordine alfabetico.