

## Sistemi Operativi – a.a. 2023/2024

prova di laboratorio  
– 19 luglio 2024 –

Creare un programma **duplicate-files.c** in linguaggio C che accetti invocazioni sulla riga di comando del tipo:

**duplicate-files** <file-1> <file-2> ... <file-n> <destination-dir>

Il programma dovrà essenzialmente creare duplicati dei file indicati nella directory indicata.

In particolare al suo avvio il programma creerà  $n+1$  thread:

- $n$  thread **READER- $i$**  che si occuperanno di leggere il file assegnato e di passarlo, un blocco alla volta, al thread **WRITER**;
- un unico thread **WRITER** che si occuperà di scrivere i blocchi ricevuti all'interno dei file duplicati nella cartella di destinazione; esso non dovrà avere accesso diretto ai file originali.

Gli  $n$  thread **READER- $i$**  agiranno in parallelo e accederanno, in sola lettura, al contenuto del rispettivo file. Il contenuto dovrà essere inviato, con blocchi di 1 KiB, attraverso record da inserire in uno stack condiviso con gli altri thread. Tale stack avrà la capienza fissa di al più 10 record e ogni record conterrà le seguenti informazioni:

- il buffer di 1024 byte per il blocco del file;
- il filename del file;
- la dimensione totale del file;
- la posizione (offset) del blocco all'interno del file (0, 1024, 2048, ...);
- la dimensione effettiva del contenuto nel buffer (potrebbe essere inferiore a 1024 byte);
- un eventuale flag di fine-lavori.

Il thread **WRITER** dovrà preliminarmente assicurarsi che la directory di destinazione esista ed in caso provvedere a crearla. Per ogni record estratto dallo stack dovrà, se non esistente, creare il file duplicato o, se esistente, a troncarlo. Successivamente dovrà scrivere il blocco ricevuto nella posizione corretta all'interno del file contenitore.

Tutti i file, in lettura o scrittura, dovranno essere utilizzati unicamente utilizzando la chiamata di sistema `mmap` per la mappatura dei file in memoria (qualunque altro metodo di accesso ai file non sarà ritenuto valido). I thread si dovranno coordinare opportunamente tramite mutex e semafori numerici POSIX: il numero (minimo) e la modalità di impiego sono da determinare da parte dello studente. Tutti i thread dovranno terminare spontaneamente al termine dei lavori.

Non si dovranno usare strutture dati con visibilità globale nel codice e si dovrà cercare di rispettare la struttura dell'output riportato nell'esempio a seguire.

**Tempo:** 2 ore e 30 minuti

La struttura dell'output atteso è la seguente:

```
$ ./duplicate-files /etc/login.defs /usr/share/dict/american-english backup

[MAIN] duplicazione di 2 file
[READER-1] lettura del file '/etc/login.defs' di 12569 byte
[READER-2] lettura del file '/usr/share/dict/american-english' di 985084 byte
[READER-1] lettura del blocco di offset 0 di 1024 byte
[READER-1] lettura del blocco di offset 1024 di 1024 byte
[READER-1] lettura del blocco di offset 2048 di 1024 byte
[READER-2] lettura del blocco di offset 0 di 1024 byte
[WRITER] creazione del file 'american-english' di dimensione 985084 byte
[WRITER] scrittura del blocco di offset 0 di 1024 byte sul file 'american-english'
[WRITER] creazione del file 'login.defs' di dimensione 12569 byte
[WRITER] scrittura del blocco di offset 2048 di 1024 byte sul file 'login.defs'
[WRITER] scrittura del blocco di offset 1024 di 1024 byte sul file 'login.defs'
[WRITER] scrittura del blocco di offset 0 di 1024 byte sul file 'login.defs'

...

[READER-1] lettura del blocco di offset 12288 di 281 byte
[READER-1] lettura del file '/etc/login.defs' completata
[READER-2] lettura del blocco di offset 984064 di 1020 byte
[READER-2] lettura del file '/usr/share/dict/american-english' completata
[WRITER] scrittura del blocco di offset 984064 di 1020 byte sul file 'american-english'
[WRITER] scrittura del file 'american-english' completata
[WRITER] scrittura del blocco di offset 12288 di 281 byte sul file 'login.defs'
[WRITER] scrittura del file 'login.defs' completata
[MAIN] duplicazione dei 2 file completata
```