

Esercizio 1

Un sistema di file sharing semplificato è costituito dalla seguente struttura di directory e file:

```
./server1/  
./server1/config.txt  
./server1/user1/  
./server1/user1/file1.part  
./server1/user1/file2.part  
./server1/user2/  
./server1/user2/file2.part  
./server1/user2/file3.part  
./server1/user3/  
./server1/user3/file1.part  
./server1/user3/file2.part  
./server1/user3/file3.part  
./server1/user4/  
./server1/user4/file4.part  
./server1/user4/file5.part  
./server1/user4/file6.part  
./server1/user5/  
./server1/user5/file6.part
```

dove:

- Le directory `user1 ... userN` rappresentano utenti del sistema di file sharing;
- I file `*.part` rappresentano le parti di un file possedute da ciascun utente, ad esempio `./server1/user1/file1.part` è una parte di un file di nome `file1`;
- Il file `config.txt` contiene informazioni sui file residenti sul `server1` insieme alle informazioni su quali parti di ciascun file sono possedute da ciascun utente.

Il file `config.txt` al suo interno è costituito da righe del seguente tipo:

```
FILE TOTDIM UNAME(NBYTE,OFFSET)
```

In cui :

- `FILE` rappresenta il nome di un file contenuto sul server, ad esempio `file1`, `file2`, etc.;
- `TOTDIM` rappresenta la dimensione in byte complessiva di `FILE`;
- `UNAME` rappresenta il nome utente che possiede `NBYTE` di `FILE` a partire da un certo `OFFSET`

Il contenuto del file `config.txt` per il server di esempio `server1` è il seguente:

```
file1 579 user1(300,0)  
file1 579 user3(279,300)  
file2 890 user1(294,293)  
file2 890 user2(293,0)
```

```
file2 890 user3(303,587)
file3 578 user2(278,300)
file3 578 user3(300,0)
file4 292 user4(292,0)
file5 276 user4(276,0)
file6 577 user4(294,283)
file6 577 user5(283,0)
```

Si scriva uno script perl `download.pl` che esegue il “download” di un file da un server; il comando deve rispettare la seguente sintassi:

```
download.pl SERVERDIR FILENAME
```

dove:

- `SERVERDIR` è il percorso in cui si trova il server, per esempio `./server1`;
- `FILENAME` è il nome del file da scaricare, ad esempio `file1`.

Lo script si deve occupare (*almeno*) di:

1. verificare se sul server esiste il file `FILENAME`;
2. individuare le parti che compongono `FILENAME`;
3. ricostruire il file in base alle informazioni ricavate dal file di configurazione del server `SERVERDIR`;
4. creare nella directory corrente il file `FILENAME`.

NOTE:

- Al fine di testare l'esercizio fare uso della directory `server1` data a corredo con la traccia d'esame.

Esercizio 2

Un sistema di file sharing semplificato è costituito dalla seguente struttura di directory e file:

```
./server2/
./server2/config.txt
./server2/file1
./server2/file2
./server2/file3
./server2/file4
./server2/file5
```

dove:

- I file `file1...fileN` rappresentano file di dati;
- Il file `config.txt` contiene informazioni sugli **utenti** del `server2` insieme alle informazioni su quali parti di ciascun file sono possedute da ciascun utente.

Il file `config.txt` al suo interno è costituito da righe del seguente tipo (come per l'Esercizio1):

```
FILE TOTDIM UNAME(NBYTE,OFFSET)
```

In cui :

- `FILE` rappresenta il nome di un file contenuto sul server, ad esempio `file1`, `file2`, etc.;
- `TOTDIM` rappresenta la dimensione in byte complessiva di `FILE`;
- `UNAME` rappresenta il nome utente che possiede `NBYTE` di `FILE` a partire da un certo `OFFSET`.

Si scriva uno script perl `share.pl` che realizza la divisione in parti dei file `file1..fileN` in base alle indicazioni contenute nel file `config.txt`; il comando deve rispettare la seguente sintassi:

```
share.pl SERVERDIR
```

dove:

- `SERVERDIR` è il percorso in cui si trova il server, per esempio `./server2`

Lo script si deve occupare (almeno) di:

1. Leggere il file di configurazione `SERVERDIR/config.txt`;
2. Per ogni riga del file nel formato `FILE TOTDIM UNAME(NBYTE,OFFSET)`:
 - a. verificare l'esistenza di `FILE` di dimensione `TOTDIM`;
 - b. creare, se non esiste la directory `UNAME`;
 - c. creare un file con nome `FILE.part` al cui interno inserire `NBYTE` di `FILE` a partire da `OFFSET`.
3. Tralasciare le righe che non rispettano il formato richiesto.

Ad esempio se leggo la riga

```
file1 579 user3(279,300)
```

devo creare in `server2` la directory `user3`, all'interno di `user3` devo creare il file `file1.part`, all'interno di `file1.part` devo inserire 279 byte di `file1` a partire dal byte 301 (compreso).

NOTE:

- Al fine di testare l'esercizio fare uso della directory `server2` data a corredo con la traccia d'esame.
- Comando `tail` - stampa a video l'ultima parte di un file:
 - o Sintassi: `tail [OPTION] filename`;
 - o Alcune `OPTION` utili:
 - `-c [+]N`: stampa a video gli ultimi `N` byte di un file; in alternativa, usando `+N`, stampa a video il contenuto di un file a partire dal byte `N`-esimo.
- Comando `head` - stampa a video la prima parte di un file:

- Sintassi: `head [OPTION] filename;`
- Alcune **OPTION** utili:
 - `-c N`: stampa a video i primi N byte di un file.