Sistemi Operativi
5 luglio 2023

Programmazione di Sistema Unix

Si progetti un'applicazione concorrente in C che consenta all'amministrazione di una biblioteca di monitorare i libri presi in prestito. L'applicazione deve presentare la seguente interfaccia:

libri_prestati dir

dove ***dir*** è un nome assoluto di directory.

Si supponga che le informazioni relative ai libri presi in prestito da un determinato utente siano salvate in un unico file nella directory ***dir***. Ad esempio, nella directory ***dir*** il file U0001.txt contiene tutte le informazioni relative ai libri presi in prestito dall'utente identificato dal codice univoco U0001. Ciascuna riga dei file contiene le informazioni di un certo libro: data di ritiro (nel formato AAAAMMGG), data di restituzione (nel formato AAAAMMGG), titolo del libro, autore e genere. Se il libro non è ancora stato restituito, allora la data di restituzione contiene la stringa "NON RESTITUITO". I campi di ciascuna riga sono separati da virgole.

L'applicazione concorrente deve essere composta da un processo **P0** che riceve dall'utente (via terminale) un identificativo di un utente e il numero N (intero) di risultati da mostrare a video. Ad esempio, se l'utente inserisce "U0001" e "5", l'applicazione deve mostrare i 5 libri presi in prestito più recentemente e non ancora restituiti dall'utente "U0001".

Per ogni richiesta inserita, il processo **P0** crea tre processi figli: **P1**, **P2** e **P3**. **P1** ordina in modo cronologico i prestiti (sulla base della data di ritiro presente nel primo campo di ogni riga) e invia tali informazioni a **P2**. **P2** seleziona solo i prestiti non ancora restituiti (sulla base del secondo campo) e invia tali informazioni a **P3**. **P3** seleziona gli N prestiti non ancora restituiti più recenti e invia tali informazioni a **P0**, che stampa tali informazioni a video.

P1 non inizia immediatamente l'ordinamento cronologico dei prestiti, ma si sospende in attesa di un segnale di sincronizzazione da parte di **P0**, che deve inviare a **P1** il segnale SIGUSR1.

L'applicazione termina quando l'utente invia un ctrl-C da terminale e, prima di terminare, stampa a video il numero di richieste servite fino a quel momento.

Sistemi Operativi
5 luglio 2023

Programmazione Shell

Si scriva uno script shell (strutturato su più file con invocazione ricorsiva e senza l'uso del comando find) per aiutare il manager di una catena di ristoranti a consultare gli ordini effettuati per un determinato piatto durante un anno specifico. In particolare, lo script deve presentare la seguente interfaccia:

ristorante piatto anno

dove ***piatto*** è un stringa alfanumerica che identifica univocamente un certo piatto servito dal ristorante e ***anno*** è un nome assoluto di directory.

Si supponga che le informazioni sugli ordini effettuati nel corso dell'anno siano memorizzate all'interno della directory ***anno*** (o in una delle sue sottodirectory) in file di testo con estensione .log. Ad esempio, le informazioni relative a tutti gli ordini effettuati il 5 luglio sono nel file 0705.log. Ciascuna riga di tali file conterrà le informazioni relative a un ordine: l'identificativo univoco del piatto (P0001, P0002, etc.), il nome del piatto e il timestamp (formato YYYYMMDDHHMM). Nota che un piatto può essere ordinato più volte nell'arco di una giornata.

Lo script ***ristorante*** deve quindi esplorare ricorsivamente la directory ***anno*** e tutte le sue sottodirectory per analizzare le informazioni presenti nei file .log, selezionare esclusivamente le informazioni relative agli ordini del piatto di interesse (argomento ***piatto***), estrarre le (sole) informazioni sul timestamp, e quindi scrivere tali dati in un file di log (chiamato piatto.log) nella home directory dell'utente. Se il file di log esiste già, lo script deve sovrascriverne il contenuto.

Prima di terminare, il file comandi deve stampare a video il contenuto del file di riepilogo piatto.log e il giorno (ovverosia il nome del file corrispondente) in cui il piatto di interesse è stato ordinato più volte.