Sistemi Operativi
13 giugno 2024

Programmazione di Sistema Unix

Si progetti un'applicazione concorrente in C per una biblioteca per gestire le richieste di prestito dei libri. L'applicazione deve presentare la seguente interfaccia:

prestiti_biblioteca dir

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

Si supponga che le informazioni sui prestiti siano salvate in file di testo con estensione ".txt", ciascuno dei quali contiene i prestiti effettuati in un mese specifico. Ad esempio, il file "202405.txt" riporta tutti i prestiti effettuati nel corso di maggio 2024. Ogni riga del file memorizza un prestito e include: data della richiesta (formato AAAAMMGG), codice del libro (stringa) e numero di tessera del cliente (stringa). I campi sono separati da virgole.

L'applicazione concorrente deve essere costituita da un processo **P0** che riceve dal bibliotecario (tramite terminale) il codice del libro di interesse (stringa), il mese di interesse (AAAAMM) e il numero N (intero maggiore di zero) di risultati da mostrare a video. Per esempio, se l'utente inserisce "B123", "202405" e "2", l'applicazione deve mostrare gli ultimi 2 prestiti relativi al libro "B123" nel corso di maggio 2024.

Per ogni richiesta inserita, il processo **P0** crea tre processi figli: **P1**, **P2** e **P3**. **P1** seleziona solo i prestiti per il libro specificato nel mese di interesse e passa queste informazioni a **P2**. **P2** ordina i prestiti in base alla data della richiesta in ordine decrescente e invia queste informazioni a **P3**. **P3** seleziona gli N prestiti più recenti e li mostra a video.

P1 non inizia immediatamente la selezione del libro, 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 dal terminale e, prima di terminare, mostra il numero totale di richieste processate fino a quel momento.

Sistemi Operativi
13 giugno 2024

Programmazione Shell

Si scriva uno script shell (strutturato su più file con invocazione ricorsiva e senza l'uso del comando find) per assistere il direttore di un'azienda di trasporti nella revisione dei viaggi effettuati durante un anno specifico. Lo script deve avere la seguente interfaccia:

azienda veicolo anno

dove ***veicolo*** è una stringa alfanumerica che identifica univocamente un tipo di veicolo (come "Autobus", "Camion", ecc.) e ***anno*** è un nome relativo di directory formato da soli caratteri numerici.

Si supponga che le informazioni sui viaggi effettuati nel corso dell'anno siano conservate all'interno della directory ***anno*** (o in una delle sue sottodirectory) in file di testo con estensione .txt. Ad esempio, i viaggi effettuati nel mese di aprile sono memorizzati in un file di testo aprile.txt. Ogni riga di tali file contiene le informazioni relative a un viaggio con, in quest'ordine: identificativo univoco del viaggio (ad esempio, T0001), la data del viaggio (nel formato YYYYMMGG), l'identificativo univoco del tipo di veicolo (Autobus, Camion, ecc.), la destinazione del viaggio e il nome dell'autista.

Lo script deve quindi esplorare ricorsivamente la directory ***anno*** e tutte le sue sottodirectory per analizzare le informazioni presenti nei file .txt, selezionare esclusivamente le informazioni relative ai viaggi della tipologia di veicolo di interesse (argomento ***veicolo***), estrarre le sole informazioni su data, destinazione e nome dell'autista e quindi scrivere tali dati in un file di log (chiamato output.txt) nella home directory dell'utente. Se il file di log esiste già, lo script deve sovrascriverne il contenuto.

Prima di terminare, lo script deve anche stampare a video il mese (ovvero il nome del file corrispondente, comprensivo di suffisso .txt) in cui sono stati effettuati più viaggi, inteso come file che ha il maggior numero di righe (a prescindere dalla tipologia di veicolo specificato).

INGLESE Copilot

Unix System Programming

Design a concurrent C application for a library to manage book loan requests. The application should present the following interface:

```
prestiti_biblioteca    dir
```

where `dir` is a relative directory name.

Assume that loan information is saved in text files with the extension ".txt," each containing loans made during a specific month. For example, the file "202405.txt" lists all loans made in May 2024. Each line in the file stores a loan and includes the following fields:

- Request date (format: AAAAMMGG)
- Book code (string)
- Customer card number (string)
- Fields are separated by commas.

The concurrent application should consist of the following processes:

1. **P0 (Parent Process)**:
    - Receives input from the librarian (via the terminal):
        - Book code of interest (string)
        - Month of interest (AAAAMM)
        - Number N (positive integer) of results to display.
    - For example, if the user enters "B123," "202405," and "2," the application should display the last 2 loans related to book "B123" in May 2024.
    - For each request, P0 creates three child processes: P1, P2, and P3.
2. **P1 (Child Process)**:
    - Selects loans for the specified book in the month of interest and passes this information to P2.
    - Waits for a synchronization signal from P0 (P0 sends the SIGUSR1 signal).
3. **P2 (Child Process)**:
    - Orders the loans based on the request date in descending order and sends this information to P3.
4. **P3 (Child Process)**:
    - Selects the N most recent loans and displays them on the screen.

P1 does not immediately start selecting the book but waits for a synchronization signal from P0, which must send the SIGUSR1 signal to P1. The application terminates when the user sends a Ctrl-C signal from the terminal. Before terminating, it shows the total number of requests processed up to that moment.

INGLESE Copilot

Shell Programming
Write a shell script (structured across multiple files with recursive invocation and without using the `find`command) to assist the director of a transportation company in reviewing trips made during a specific year. The script should have the following interface:

```
azienda     veicolo     anno
```

where:

- `azienda` represents the company.
- `veicolo` is an alphanumeric string that uniquely identifies a type of vehicle (such as “Autobus,” “Camion,” etc.).
- `anno` is a relative directory name composed solely of numeric characters.

Assume that information about trips made during the year is stored within the `anno` directory (or its subdirectories) in text files with the extension `.txt`. For example, trips made in April are stored in a text file named `aprile.txt`. Each line in these files contains the following information about a trip, in order:

- Unique trip identifier (e.g., T0001)
- Trip date (in the format YYYYMMDD)
- Unique identifier of the vehicle type (Autobus, Camion, etc.)
- Destination of the trip
- Driver’s name

The script should recursively explore the `anno` directory and its subdirectories to analyze the information present in the `.txt` files. It should selectively extract information related to trips of the specified vehicle type (specified by the `veicolo` argument), including date, destination, and driver’s name. The script should then write this data to a log file named `output.txt` in the user’s home directory. If the log file already exists, the script should overwrite its content. Before terminating, the script should also print to the screen the month (i.e., the name of the corresponding file, including the `.txt` suffix) in which the most trips were made. This is determined by the file with the highest number of lines, regardless of the specified vehicle type.