POOL DI THREAD

LIBRERIE NECESSARIE

- `pthread.h`: La libreria principale per gestire i thread in C.
- `stdio.h`: Per la gestione dell'input/output, utile per i log e i messaggi di errore.
- `stdlib.h`: Per la gestione della memoria dinamica e altre funzioni generiche.
- `unistd.h`: Contiene funzioni come `sleep`, `close`, che potrebbero esserti utili per gestire i socket.
- `string.h`: Per manipolare stringhe, utile per operazioni come la copia e il confronto di stringhe.

STRUTTURE DATI

Thread pool

- `pthread_t* threads`: Un array di thread che rappresentano il pool di thread.
- `int available_threads`: Un contatore che tiene traccia dei thread disponibili per gestire nuove richieste.
- `pthread_mutex_t lock`: Un mutex per sincronizzare l'accesso al contatore di thread disponibili.
- `pthread_cond_t cond`: Una variabile di condizione per far "dormire" i thread quando non ci sono richieste da gestire.

FUNZIONI

1. **Funzione per Gestire la Connessione Client. Spiegazione**: Gestisce la comunicazione con un singolo client, utilizzando un socket dedicato per quella connessione.

```
void* handle_client(void* arg);
```

2. **Funzione per il Thread del Pool. Spiegazione**: Ogni thread del pool esegue questa funzione. Attende che ci sia una richiesta da gestire (una connessione client). Quando c'è una richiesta, il thread si occupa della comunicazione con il client tramite la funzione handle_client.

```
void* thread_function(void* arg);
```

3. **Funzione per Inizializzare il Pool di Thread Spiegazione**: Inizializza il pool di thread, allocando memoria per i thread e configurando mutex e variabili di condizione. Crea i thread nel pool, che sono pronti a gestire le connessioni.

```
void init_thread_pool(ThreadPool* pool);
```

4. **Funzione per Distruggere il Pool di Thread Spiegazione**: Distrugge il pool di thread, liberando la memoria allocata per i thread, mutex e variabili di condizione, una volta che il server ha finito di gestire

le connessioni.

```
void destroy_thread_pool(ThreadPool* pool);
```

5. **Funzione per il Ciclo Principale del Server Spiegazione**: Avvia il server, crea il socket di ascolto, accetta le connessioni dai client e le assegna ai thread del pool. Alla fine, chiude il server e distrugge il pool di thread.

```
int main();
```

CONCETTI CHIAVE

- **Creazione di un socket per ogni client:** Ogni client ha un socket dedicato per la sua connessione, gestito dal thread che si occupa della comunicazione con il client.
- **Thread Pool:** Un insieme di thread che aspettano connessioni da gestire. Quando un client si connette, un thread del pool prende in carico la connessione.
- **Sincronizzazione:** Il mutex e la variabile di condizione vengono utilizzati per sincronizzare l'accesso alle risorse condivise (come il numero di thread disponibili) e per coordinare il comportamento dei thread.

FLUSSO DI ESECUZIONE