Temă: Bibliotecă pentru Manipularea Fișierelor Multi-Platformă

Curs de Sisteme de Operare

December 19, 2024

Introducere

Biblioteca standard C oferă o abstractizare pentru lucrul cu fișiere prin funcții precum `fopen()`, `fread()`, `fwrite()` și `fclose()`. Aceste funcții simplifică manipularea fișierelor și asigură consistență între diferite sisteme de operare, ascunzând detalii de implementare specifice platformei. În această temă, veți avea rolul de a construi o abstractizare similară: o bibliotecă multi-platformă pentru manipularea fișierelor, `so_file_lib`, care să funcționeze fără întreruperi atât pe Linux, cât și pe Windows.

Această sarcină este un exercițiu valoros pentru înțelegerea modului în care sunt implementate straturile de abstractizare și cum funcționează API-urile sistemului de operare la nivel intern. La finalul temei, veți avea o înțelegere mai profundă asupra modului în care biblioteci precum cea standard C oferă astfel de facilități.

Descrierea Sarcinii

Folosind fișierele schelet furnizate, implementați următoarele funcții în `so_file_lib.c`:

1. `SO_FILE *so_fopen(const char *pathname, const char *mode);`
2. `int so_fclose(SO_FILE *stream);`
3. `size_t so_fread(void *ptr, size_t size, size_t count, SO_FILE *stream);`
4. `size_t so_fwrite(const void *ptr, size_t size, size_t count, SO_FILE *stream);`

5. `int so_fseek(SO_FILE *stream, long offset, int whence);`

6. `long so_ftell(SO_FILE *stream);`

Trebuie să asigurați compatibilitatea atât cu sistemele Linux, cât și cu cele Windows. Folosiți directiva `#ifdef` pentru a include antete specifice platformei și pentru a implementa apelurile de API corespunzătoare.

Documentație Furnizată

Ca referință, iată un rezumat al principalelor apeluri de sistem și a echivalentelor lor pe diferite platforme:

- **Deschidere fișier:** `open()` (Linux) vs `CreateFile()` (Windows).
- **Citire fișier:** `read()` (Linux) vs `ReadFile()` (Windows).
- **Scriere fișier:** `write()` (Linux) vs `WriteFile()` (Windows).
- **Poziționare în fișier:** `lseek()` (Linux) vs `SetFilePointer()` (Windows).
- **Închidere fișier:** `close()` (Linux) vs `CloseHandle()` (Windows).

Fiecare dintre aceste API-uri are particularitățile sale.

Prezentare a Instrumentelor de Dezvoltare

Veți avea nevoie de instrumente pentru a compila și testa codul atât pe Linux, cât și pe Windows. Mai jos este o scurtă prezentare a instrumentelor necesare:

- **GCC (GNU Compiler Collection):** Un compilator popular pe Linux pentru programe C/C++, care vă permite să compilați codul în executabile.
- **CMake:** Un generator de sisteme de build multi-platformă. Ajută la asigurarea faptului că proiectul se compilează corect atât pe Linux, cât și pe Windows, indiferent de compilatorul de bază.
- **Microsoft Visual Studio:** Cel mai utilizat IDE pe Windows, care include compilatorul MSVC (`cl.exe`).

Testare și Verificare

Pentru a verifica corectitudinea, a fost furnizat un script utilitar de testare și compilare, `helper.py`, care include următoarele funcționalități:

- **Compilare:** Utilizați comanda `python3 helper.py --compile` pentru a compila soluția folosind CMake. Aceasta asigură că codul se construiește corect atât pe Linux, cât și pe Windows.
- **Testare:** Utilizați `python3 helper.py --check` pentru a rula o serie de teste predefinite care validează implementarea.

Testele vor verifica faptul că funcțiile dumneavoastră funcționează conform așteptărilor, incluzând deschiderea fișierelor, citirea, scrierea, poziționarea și închiderea lor. Folosiți scriptul pentru a vă verifica soluția înainte de a o trimite.

Instrucțiuni de Trimitere

Trimiteți o arhivă care conține fișierul completat `so_file_lib.c`.

Criterii de Evaluare

Evaluarea soluției se va face pe baza:

- Corectitudinii: Implementarea funcțiilor atât pentru Linux, cât și pentru Windows.
- Testării: Dovada că implementarea funcționează corect pe ambele platforme.
- Calității codului: Claritatea, utilizarea corectă a comentariilor și respectarea standardelor de codare.
- Siguranței codului: Asigurați-vă că nu există scurgeri de memorie la ieșirea din program. Puteți verifica acest lucru folosind `valgrind`.

Referințe

- Documentație API Linux: `https://man7.org/linux/man-pages/`

- Documentație API Windows: `https://learn.microsoft.com/en-us/windows/win32/api/`
- GCC: `https://gcc.gnu.org/`
- Microsoft Visual Studio: `https://visualstudio.microsoft.com/`
- CMake: `https://cmake.org/`
- Valgrind: `https://valgrind.org/`