Робота з файлами 2. Тема 13

Ліна Явдошак

1 грудня 2024 р.

Мета проєкту

Метою даного проєкту є розробка програмного забезпечення для автоматизації роботи з файлами та директоріями. Програма повинна виконувати наступні завдання:

1. Зміна розширення файлів з *.c на *.cpp.
2. Заміна однорядкових коментарів у файлах з розширенням *.cpp.
3. Видалення текстових файлів, які були модифіковані раніше заданої дати.
4. Перенесення текстових файлів, створених раніше ніж рік тому, в іншу директорію.
5. Видалення файлів з розширеннями Word, які менше 100 кб.
6. Обчислення середнього розміру текстових файлів у заданій директорії.

Опис та пояснення бібліотек та функцій

Бібліотека <filesystem> (C++17)

Бібліотека `<filesystem>` є частиною стандарту C++17 і надає можливості для роботи з файловою системою. Вона включає в себе класи та функції для маніпуляції шляхами, перегляду директорій, отримання інформації про файли та директорії.

Основні функції та класи

- `std::filesystem::path` - клас для роботи з шляхами до файлів та директорій.
- `std::filesystem::directory_iterator` - ітератор для перегляду вмісту директорії.
- `std::filesystem::recursive_directory_iterator` - ітератор для рекурсивного перегляду вмісту директорії та її піддиректорій.
- `std::filesystem::rename` - функція для перейменування або переміщення файлу.
- `std::filesystem::remove` - функція для видалення файлу.
- `std::filesystem::file_size` - функція для отримання розміру файлу.
- `std::filesystem::last_write_time` - функція для отримання часу останньої модифікації файлу.

Бібліотека <dirent.h> (C)

Бібліотека `<dirent.h>` є стандартною бібліотекою мови C і надає можливості для роботи з директоріями. Вона включає в себе функції для відкриття, читання та закриття директорій, а також для отримання інформації про файли та директорії.

Основні функції та структури

- `DIR` - структура, що представляє директорію.
- `opendir` - функція для відкриття директорії.
- `readdir` - функція для читання наступного запису з директорії.
- `closedir` - функція для закриття директорії.
- `struct dirent` - структура, що містить інформацію про файл або директорію.

Детальний опис використаних функцій для кожного завдання

Завдання 1: Зміна розширення файлів з *.c на *.cpp

```
static void change_c_to_cpp(const char* file_path) {
    if (strstr(file_path, ".c") && !strstr(file_path
       ↪ , ".cpp")) {
        char new_path[MAX_PATH];
        snprintf(new_path, MAX_PATH, "%.*s.cpp", (
           ↪ int)(strlen(file_path) - 2), file_path)
           ↪ ;
        if (rename(file_path, new_path) == 0) {
            printf("Renamed: %s -> %s\n", file_path,
               ↪  new_path);
        }
        else {
            perror("rename");
        }
    }
}
```

Ця функція перевіряє, чи має файл розширення `.c` і не має розширення `.cpp`. Якщо умова виконується, вона формує новий шлях з розширенням `.cpp` та перейменовує файл за допомогою функції `rename`.

Завдання 2: Заміна однорядкових коментарів у файлах з розширенням *.cpp

```
static void replace_single_line_comments(const char*
   ↪  file_path) {
    if (strstr(file_path, ".cpp")) {
        FILE* file = fopen(file_path, "r");
        if (!file) {
            perror("fopen");
            return;
        }

        char temp_path[MAX_PATH];
```

```
            snprintf(temp_path, MAX_PATH, "%s.tmp",
               ↪ file_path);

            FILE* temp_file = fopen(temp_path, "w");
            if (!temp_file) {
                perror("fopen");
                fclose(file);
                return;
            }

            char line[MAX_PATH];
            while (fgets(line, MAX_PATH, file)) {
                char* comment_pos = strstr(line, "//");
                if (comment_pos) {
                    *comment_pos = '\0';
                    fprintf(temp_file, "%s/* %s */\n",
                       ↪ line, comment_pos + 2);
                }
                else {
                    fputs(line, temp_file);
                }
            }

            fclose(file);
            fclose(temp_file);

            if (rename(temp_path, file_path) != 0) {
                perror("rename");
            }
        }
    }
```

Ця функція відкриває файл з розширенням `.cpp`, шукає однорядкові коментарі (`//`) та замінює їх на багаторядкові коментарі (`/* ... */`). Для цього вона створює тимчасовий файл, записує в нього модифікований вміст, а потім перейменовує тимчасовий файл на оригінальний.

Завдання 3: Видалення текстових файлів, які були модифіковані раніше заданої дати

```
static void delete_txt_older_than_date(const char*
   ↪ file_path, time_t cutoff_date) {
    if (strstr(file_path, ".txt")) {
        struct stat sb;
        if (stat(file_path, &sb) == 0 && sb.st_mtime
           ↪  < cutoff_date) {
            if (remove(file_path) == 0) {
                printf("Deleted: %s\n", file_path);
            }
            else {
                perror("remove");
            }
        }
    }
}
```

Ця функція перевіряє, чи має файл розширення `.txt` та чи був він модифікований раніше заданої дати. Якщо умова виконується, файл видаляється за допомогою функції `remove`.

Завдання 4: Перенесення текстових файлів, створених раніше ніж рік тому, в іншу директорію

```
static void move_txt_older_than_year(const char*
   ↪ file_path, const char* target_dir, time_t
   ↪ cutoff_date) {
    if (strstr(file_path, ".txt")) {
        struct stat sb;
        if (stat(file_path, &sb) == 0 && sb.st_ctime
           ↪  < cutoff_date) {
            char new_path[MAX_PATH];
            snprintf(new_path, MAX_PATH, "%s/%s",
               ↪ target_dir, strrchr(file_path, '/')
               ↪  + 1);
            if (rename(file_path, new_path) == 0) {
                printf("Moved: %s -> %s\n",
                   ↪ file_path, new_path);
            }
            else {
                perror("rename");
            }
```

```
        }
    }
}
```

Ця функція перевіряє, чи має файл розширення `.txt` та чи був він створений раніше ніж рік тому. Якщо умова виконується, файл переміщується в іншу директорію за допомогою функції `rename`.

Завдання 5: Видалення файлів з розширеннями Word, які менше 100 кб

```
static void delete_word_files_small(const char*
    ↪ file_path) {
    if (strstr(file_path, ".doc") || strstr(
        ↪ file_path, ".docx")) {
        struct stat sb;
        if (stat(file_path, &sb) == 0 && sb.st_size
            ↪ < 100 * 1024) {
            if (remove(file_path) == 0) {
                printf("Deleted: %s\n", file_path);
            }
            else {
                perror("remove");
            }
        }
    }
}
```

Ця функція перевіряє, чи має файл розширення `.doc` або `.docx` та чи менше він 100 кб. Якщо умова виконується, файл видаляється за допомогою функції `remove`.

Завдання 6: Обчислення середнього розміру текстових файлів у заданій директорії

```
static void calculate_average_txt_size(const char*
    ↪ path) {
    struct dirent* entry;
    DIR* dp = opendir(path);
    if (!dp) {
        perror("opendir");
```

```
        return;
    }

    size_t total_size = 0, file_count = 0;

    while ((entry = readdir(dp))) {
        if (strcmp(entry->d_name, ".") == 0 ||
            ↪ strcmp(entry->d_name, "..") == 0)
            continue;

        char full_path[MAX_PATH];
        snprintf(full_path, MAX_PATH, "%s/%s", path,
            ↪  entry->d_name);

        struct stat sb;
        if (stat(full_path, &sb) == 0 && S_ISREG(sb.
            ↪ st_mode) && strstr(entry->d_name, ".txt
            ↪ ")) {
            total_size += sb.st_size;
            file_count++;
        }
    }

    closedir(dp);

    if (file_count > 0) {
        printf("Average size: %zu bytes\n",
            ↪ total_size / file_count);
    }
    else {
        printf("No text files found.\n");
    }
}
```

Ця функція обчислює середній розмір текстових файлів (`.txt`) у заданій директорії. Вона відкриває директорію, переглядає всі файли, підсумовує їх розміри та обчислює середнє значення.

Висновок

У цьому проєкті ми розробили програмне забезпечення для автоматизації роботи з файлами та директоріями. Ми використали бібліотеки `<filesystem>` (C++17) та `<dirent.h>` (C) для виконання різних завдань, таких як зміна розширення файлів, заміна коментарів, видалення та переміщення файлів, а також обчислення середнього розміру файлів. Програма демонструє можливості цих бібліотек та ефективність їх використання для роботи з файловою системою.