Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІКТА

Кафедра СКС



Лабораторна робота № 4

З дисципліни «Системне програмне забезпечення*»*

На тему: «Робота з файлами та файловими системами»

Виконала:

ст. гр. КІ-307

Ващишин І.І.

Прийняв:

ст. викладач

Олексів М. В.

Львів – 2025

**Мета:** Навчитися розробляти програми реалізації операцій з файлами та каталогами.

**Теоретичні відомості:**

**Загальні відомості про файлові операції:**

Основні файлові операції у прикладних програмах:

**Відкриття файлу.** Після відкриття файлу процес може із ним працювати (робити читання і записування). Операція передбачає завантаження в ОЗП спеціальної структури даних — **дескриптора файлу**, який визначає його атрибути та місце розташування на диску. Наступні виклики використовуватимуть цю структуру для доступу до файлу.

**Закриття файлу.** Структуру даних, створену під час його відкриття, вилучають із пам'яті. Усі не збережені зміни записують на диск.

**Створення файлу.** Операція створює на диску новий файл нульової довжини. Створений файл автоматично відкривають.

**Вилучення файлу.** Операція вилучає файл і вивільняє зайнятий ним дисковий простір. Вона недопустима для відкритих файлів.

**Читання з файлу.** Операція пересилає певну кількість байтів із файлу, починаючи із поточної позиції, у заздалегідь виділений для цього буфер пам'яті режиму користувача.

**Записування у файл.** Дані із поточної позиції записують у файл із заздалегідь виділеного буфера. Якщо на цій позиції вже є дані, вони будуть перезаписані.

**Переміщення покажчика поточної позиції.** Перед операціями читання і записування слід визначити, де у файлі перебувають потрібні дані або куди треба їх записати, задавши за допомогою цієї операції поточну позицію у файлі.

**Отримання і завдання атрибутів файлу.** Ці дві операції дають змогу зчитувати поточні значення всіх або деяких атрибутів файлу або задавати для них нові значення.

**Операції з каталогами:**

1. **Створення нового каталогу.** Операція створює новий каталог.
2. **Вилучення каталогу.** На рівні системного виклику операція дозволена тільки для порожніх каталогів.
3. **Відкриття і закриття каталогу.** Каталог має бути відкритий перед використанням і закритий після використання.
4. **Читання елемента каталогу.** Операція зчитує один елемент каталогу і переміщує поточну позицію на наступний елемент. В циклі можна обійти весь каталог.
5. **Перехід у початок каталогу.** Операція переміщує поточну позицію до першого елемента каталогу.

**Хід роботи:**

**Завдання:**

Розробити програму, в якій будуть реалізовані наступні функції:

* видалення файлів;
* перевірити чи файл прихований;
* визначити час і дату створення каталогу.

1. **Основні моменти коду:**

* **Перевірка існування файлу:**

BOOL fileExists(const char\* path) {

DWORD attr = GetFileAttributesA(path);

return (attr != INVALID\_FILE\_ATTRIBUTES && !(attr & FILE\_ATTRIBUTE\_DIRECTORY));

}

Суть: Перевіряє, чи файл існує і не є директорією.

* **Видалення файлу:**

void deleteFile(const char\* path) {

if (!fileExists(path)) {

printf("Файл не існує: %s\n", path);

return;

}

if (DeleteFileA(path)) {

printf("Файл успішно видалено: %s\n", path);

}

else {

printf("Помилка при видаленні файлу.\n");

}

}

**Суть:** перевіряє, чи файл існує, і якщо так — намагається його видалити через DeleteFileA.

* **Перевірка чи файл прихований:**

void checkIfHidden(const char\* path) {

if (!fileExists(path)) {

printf("Файл не існує: %s\n", path);

return;

}

DWORD attributes = GetFileAttributesA(path);

printf("Файл %s прихований: %s\n", path, (attributes & FILE\_ATTRIBUTE\_HIDDEN) ? "Так" : "Ні");

}

Суть: перевіряє атрибути файлу й виводить, чи він має прапорець HIDDEN.

* **Отримання часу створення каталогу:**

void printCreationTime(const char\* directoryPath) {

HANDLE hDir = CreateFileA(

directoryPath,

GENERIC\_READ,

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE | FILE\_SHARE\_DELETE,

NULL,

OPEN\_EXISTING,

FILE\_FLAG\_BACKUP\_SEMANTICS,

NULL

);

if (hDir == INVALID\_HANDLE\_VALUE) {

printf("Не вдалося відкрити каталог.\n");

return;

}

FILETIME creationTime;

if (GetFileTime(hDir, &creationTime, NULL, NULL)) {

SYSTEMTIME stUTC, stLocal;

FileTimeToSystemTime(&creationTime, &stUTC);

SystemTimeToTzSpecificLocalTime(NULL, &stUTC, &stLocal);

printf("Час створення каталогу %s: %02d.%02d.%04d %02d:%02d:%02d\n",

directoryPath,

stLocal.wDay, stLocal.wMonth, stLocal.wYear,

stLocal.wHour, stLocal.wMinute, stLocal.wSecond

);

}

else {

printf("Не вдалося отримати час створення каталогу.\n");

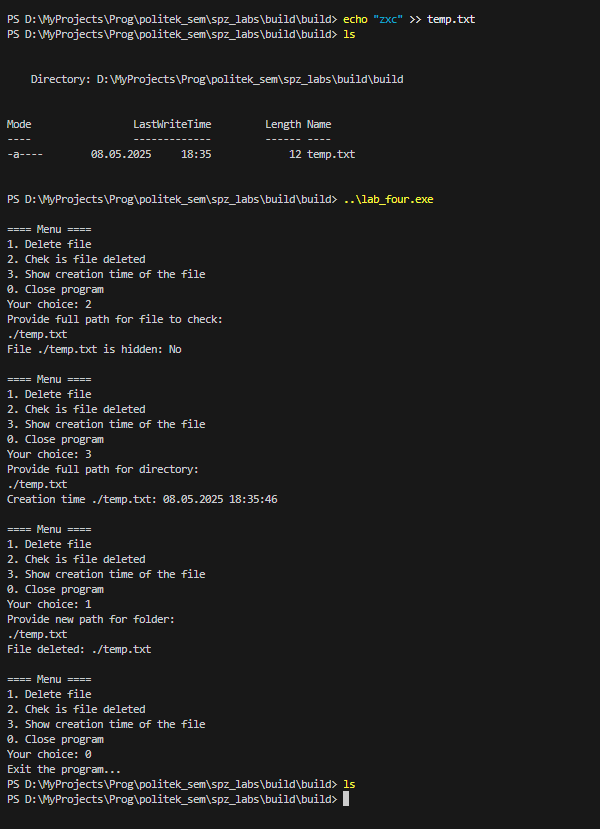
}

CloseHandle(hDir);

}

Суть: відкриває каталог як об'єкт файлу й виводить його час створення, перетворюючи його у локальний час.

**Перевірка роботи програми:**



**Висновок:** на даній лабораторній роботі я навчилася розробляти програми реалізації операцій з файлами та каталогами. У ході лабораторної роботи була розроблена програма, в якій були реалізовані наступні функції:

* видалення файлів;
* перевірити чи файл прихований;
* визначити час і дату створення каталогу.

Програма працює успішно, виконує всі поставлені вимоги.

**Додаток:**

**Код програми.**

#include <stdio.h>

#include <windows.h>

#include <locale.h>

BOOL fileExists(const char\* path) {

DWORD attr = GetFileAttributesA(path);

return (attr != INVALID\_FILE\_ATTRIBUTES && !(attr & FILE\_ATTRIBUTE\_DIRECTORY));

}

void deleteFile(const char\* path) {

if (!fileExists(path)) {

printf("File doesnt exist: %s\n", path);

return;

}

if (DeleteFileA(path)) {

printf("File deleted: %s\n", path);

}

else {

printf("Error in file deletion.\n");

}

}

void checkIfHidden(const char\* path) {

if (!fileExists(path)) {

printf("File doesnt exist: %s\n", path);

return;

}

DWORD attributes = GetFileAttributesA(path);

printf("File %s is hidden: %s\n", path, (attributes & FILE\_ATTRIBUTE\_HIDDEN) ? "Yes" : "No");

}

void printCreationTime(const char\* directoryPath) {

HANDLE hDir = CreateFileA(

directoryPath,

GENERIC\_READ,

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE | FILE\_SHARE\_DELETE,

NULL,

OPEN\_EXISTING,

FILE\_FLAG\_BACKUP\_SEMANTICS,

NULL

);

if (hDir == INVALID\_HANDLE\_VALUE) {

printf("Cannot open file.\n");

return;

}

FILETIME creationTime;

if (GetFileTime(hDir, &creationTime, NULL, NULL)) {

SYSTEMTIME stUTC, stLocal;

FileTimeToSystemTime(&creationTime, &stUTC);

SystemTimeToTzSpecificLocalTime(NULL, &stUTC, &stLocal);

printf("Creation time %s: %02d.%02d.%04d %02d:%02d:%02d\n",

directoryPath,

stLocal.wDay, stLocal.wMonth, stLocal.wYear,

stLocal.wHour, stLocal.wMinute, stLocal.wSecond

);

}

else {

printf("Cannot update creation time.\n");

}

CloseHandle(hDir);

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

char filePath[MAX\_PATH];

char dirPath[MAX\_PATH];

int choice;

do {

printf("\n==== Menu ====\n");

printf("1. Delete file\n");

printf("2. Chek is file deleted\n");

printf("3. Show creation time of the file\n");

printf("0. Delete from program\n");

printf("Your choice: ");

scanf("%d", &choice);

getchar();

switch (choice) {

case 1:

printf("Provide new path for folder:\n");

fgets(filePath, MAX\_PATH, stdin);

filePath[strcspn(filePath, "\n")] = '\0';

deleteFile(filePath);

break;

case 2:

printf("Provide full path for file to check:\n");

fgets(filePath, MAX\_PATH, stdin);

filePath[strcspn(filePath, "\n")] = '\0';

checkIfHidden(filePath);

break;

case 3:

printf("Provide full path for directory:\n");

fgets(dirPath, MAX\_PATH, stdin);

dirPath[strcspn(dirPath, "\n")] = '\0';

printCreationTime(dirPath);

break;

case 0:

printf("Exit the program...\n");

break;

default:

printf("Wrong choice. try againx.\n");

}

} while (choice != 0);

return 0;

}