Практичне завдання  
з курсу системне програмне забезпечення

Виконав: ст.гр КІ-302

Козак О.П.

Перевірив: Гаваньо Б.І.

Завдання:

Розробити дві програми. Перша шукає кількість малих латинських літер у текстовому файлі. Пошук кількісті малих латинських літер оформити як функцію потоку. Друга програма запускає першу як новостворений процес. Обидві програми мають виводити інформацію про усі запущені процеси і потоки (дескриптор та ідентифікатор).

***Програма 1***

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

struct ThreadArgs {

LPCWSTR filename;

int lowercase\_count;

};

DWORD WINAPI countLowercaseLetters(LPVOID lpParam) {

struct ThreadArgs\* threadArgs = (struct ThreadArgs\*)lpParam;

LPCWSTR filename = threadArgs->filename;

int count = 0;

HANDLE hFile = CreateFile(filename, GENERIC\_READ, FILE\_SHARE\_READ, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

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

fprintf(stderr, "Failed to open file (%lu).\n", GetLastError());

return 1;

}

CHAR buffer[1024];

DWORD bytesRead;

while (ReadFile(hFile, buffer, sizeof(buffer), &bytesRead, NULL) && bytesRead > 0) {

for (DWORD i = 0; i < bytesRead; i++) {

if (buffer[i] >= 'a' && buffer[i] <= 'z') {

count++;

}

}

}

CloseHandle(hFile);

threadArgs->lowercase\_count = count;

return 0;

}

int main() {

LPCWSTR filename = L"C:\\Study\\302\_2\\SPZ\\exam.txt";

struct ThreadArgs threadArgs;

threadArgs.filename = filename;

threadArgs.lowercase\_count = 0;

HANDLE hThread;

DWORD dwThreadId;

hThread = CreateThread(

NULL,

0,

countLowercaseLetters,

&threadArgs,

0,

&dwThreadId);

if (hThread == NULL) {

fprintf(stderr, "Error creating thread (%lu).\n", GetLastError());

return 1;

}

WaitForSingleObject(hThread, INFINITE);

printf("\nNumber of lowercase letters in the file: %d\n", threadArgs.lowercase\_count);

printf("\nThread ID: %lu\n", dwThreadId);

printf("Thread Handle: %p\n\n", hThread);

printf("Main process ID: %lu\n", GetCurrentProcessId());

printf("Main thread ID: %lu\n", GetCurrentThreadId());

printf("Main process Handle: %p\n", GetCurrentProcess());

printf("Main thread Handle: %p\n", GetCurrentThread());

CloseHandle(hThread);

getchar();

return 0;

}

***Програма 2***

#include <stdio.h>

#include <windows.h>

int main() {

LPCWSTR lpApplicationName = L"C:\\Study\\302\_2\\SPZ\\lab1\_1\\x64\\Debug\\lab1\_1.exe";

PROCESS\_INFORMATION pi;

STARTUPINFO si;

int status = 0;

ZeroMemory(&si, sizeof(si));

si.cb = sizeof(si);

ZeroMemory(&pi, sizeof(pi));

si.dwFlags = STARTF\_USESHOWWINDOW;

si.wShowWindow = SW\_SHOWDEFAULT;

if (!CreateProcess(

lpApplicationName,

NULL,

NULL,

NULL,

FALSE,

CREATE\_NEW\_CONSOLE,

NULL,

NULL,

&si,

&pi)) {

fprintf(stderr, "CreateProcess failed (%lu).\n", GetLastError());

return 1;

}

printf("Child process started...\n");

printf("\nChild process ID: %lu\n", pi.dwProcessId);

printf("Child thread ID: %lu\n", pi.dwThreadId);

printf("Child process Handle: %p\n", pi.hProcess);

printf("Child thread Handle: %p\n", pi.hThread);

printf("\nParent process ID: %lu\n", GetCurrentProcessId());

printf("Parent thread ID: %lu\n", GetCurrentThreadId());

printf("Parent process Handle: %p\n", GetCurrentProcess());

printf("Parent thread Handle: %p\n", GetCurrentThread());

WaitForSingleObject(pi.hProcess, INFINITE);

GetExitCodeProcess(pi.hProcess, (LPDWORD)&status);

printf("\nChild process finished with status %d\n", status);

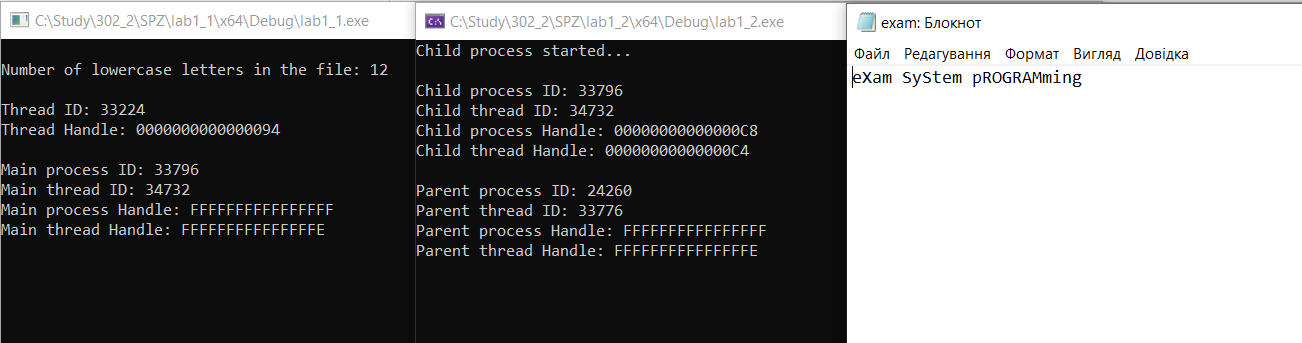
CloseHandle(pi.hProcess);

CloseHandle(pi.hThread);

return 0;

}

Результати роботи програми:



Обидві програми демонструють роботу з потоками (threads) та процесами (processes) у Windows. Ось короткий опис їх роботи:

**Перша програма**

Перша програма читає файл та рахує кількість малих літер у ньому за допомогою окремого потоку.

1. **Структура даних**: struct ThreadArgs містить два поля: ім'я файлу та кількість малих літер.
2. **Функція потоку**: countLowercaseLetters - функція, яка виконується в окремому потоці. Вона відкриває файл, читає його вміст блоками та рахує кількість малих літер у цих блоках. Після завершення роботи результат записується у структуру ThreadArgs.
3. **Основна функція**:

Ініціалізує структуру ThreadArgs.

Створює новий потік, що виконує countLowercaseLetters.

Чекає завершення потоку за допомогою WaitForSingleObject.

Виводить кількість малих літер та інформацію про потік і процес.

**Друга програма**

Друга програма створює новий процес для запуску іншої програми та очікує його завершення.

1. Ініціалізація: Ініціалізує структури STARTUPINFO та PROCESS\_INFORMATION.
2. Запуск процесу:

Використовує CreateProcess для створення нового процесу з зазначеним ім'ям виконуваного файлу (lpApplicationName).

Якщо створення процесу успішне, виводить інформацію про дочірній процес (його ID, дескриптори) та інформацію про батьківський процес.

1. Очікування завершення:

Використовує WaitForSingleObject для очікування завершення дочірнього процесу.

Після завершення дочірнього процесу отримує його код виходу за допомогою GetExitCodeProcess.

Виводить код виходу дочірнього процесу та закриває дескриптори процесу та потоку.

Обидві програми демонструють основні принципи роботи з потоками та процесами в Windows, зокрема створення, управління та синхронізацію.