Министерство науки и высшего образования Российской Федерации Пензенский государственный университет

Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №7

по дисциплине «Основы операционных систем»

на тему «Работа с файлами в асинхронном режиме»

Выполнили студенты группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Захаров А.С.

Приняли:

Егоров В. Ю.

Федюнин Р. Н.

Пенза 2024

**Название**

Работа с файлами в асинхронном режиме

**Цель работы**

Изучение способов работы с файлами в Win32 API. Изучение асинхронного режима работы с файлами.

**Лабораторное задание**

Разработать программу, осуществляющую транслитерацию символов русского алфавита в текстовом файле в символы латинского алфавита. Чтение из файла, запись в файл и транслитерация должны осуществляться параллельно подсчету. Процесс программы должен состоять из одной нити.

**Описание данных**

* input.txt — входной файл с текстом для транслитерации.
* output.txt — выходной файл для сохранения транслитерированного текста.
* buffer1 и buffer2 (по 4090 байтов) для чтения данных и хранения результата транслитерации.
* hEventRead и hEventWrite для синхронизации операций чтения и записи.
* bytesRead, bytesWritten — количество считанных/записанных байтов.
* OVERLAPPED структуры для асинхронного ввода/вывода.
* transliterateChar заменяет кириллицу на латиницу, оставляя остальные символы без изменений.

**Текст программы**

#include <windows.h>

#include <stdio.h>

#include <stdbool.h>

#include <locale.h>

#define BUFFER\_SIZE 4090

char transliterateСhar(char c) {

switch (c) {

case 'а': return 'a';

case 'б': return 'b';

case 'в': return 'v';

case 'г': return 'g';

case 'д': return 'd';

case 'е': return 'e';

case 'ё': return 'e';

case 'ж': return 'z';

case 'з': return 'z';

case 'и': return 'i';

case 'й': return 'y';

case 'к': return 'k';

case 'л': return 'l';

case 'м': return 'm';

case 'н': return 'n';

case 'о': return 'o';

case 'п': return 'p';

case 'р': return 'r';

case 'с': return 's';

case 'т': return 't';

case 'у': return 'u';

case 'ф': return 'f';

case 'х': return 'h';

case 'ц': return 'c';

case 'ч': return 'c';

case 'ш': return 's';

case 'щ': return 's';

case 'ъ': return '\'';

case 'ы': return 'y';

case 'ь': return '\'';

case 'э': return 'e';

case 'ю': return 'u';

case 'я': return 'y';

case 'А': return 'A';

case 'Б': return 'B';

case 'В': return 'V';

case 'Г': return 'G';

case 'Д': return 'D';

case 'Е': return 'E';

case 'Ё': return 'E';

case 'Ж': return 'Z';

case 'З': return 'Z';

case 'И': return 'I';

case 'Й': return 'Y';

case 'К': return 'K';

case 'Л': return 'L';

case 'М': return 'M';

case 'Н': return 'N';

case 'О': return 'O';

case 'П': return 'P';

case 'Р': return 'R';

case 'С': return 'S';

case 'Т': return 'T';

case 'У': return 'U';

case 'Ф': return 'F';

case 'Х': return 'H';

case 'Ц': return 'C';

case 'Ч': return 'C';

case 'Ш': return 'S';

case 'Щ': return 'S';

case 'Ъ': return '\'';

case 'Ы': return 'Y';

case 'Ь': return '\'';

case 'Э': return 'E';

case 'Ю': return 'U';

case 'Я': return 'Y';

default: return c;

}

}

void transliterateBuffer(char\* buffer, char\* resBuffer, DWORD bytesRead) {

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

resBuffer[i] = transliterateСhar(buffer[i]);

}

}

int main() {

HANDLE hSource, hDest;

OVERLAPPED overlappedRead = { 0 }, overlappedWrite = { 0 };

HANDLE hEventRead, hEventWrite;

DWORD bytesRead = 0, bytesWritten = 0;

char buffer1[BUFFER\_SIZE], buffer2[BUFFER\_SIZE];

char\* currentBuffer = buffer1;

char\* processingBuffer = buffer2;

hEventRead = CreateEvent(NULL, TRUE, FALSE, NULL);

hEventWrite = CreateEvent(NULL, TRUE, FALSE, NULL);

if (!hEventRead || !hEventWrite) {

fprintf(stderr, "Failed to create events.\n");

return 1;

}

hSource = CreateFileW(L"input.txt", GENERIC\_READ, 0, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

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

fprintf(stderr, "Failed to open input file.\n");

return 1;

}

hDest = CreateFileW(L"output.txt", GENERIC\_WRITE, 0, NULL, CREATE\_ALWAYS, FILE\_FLAG\_OVERLAPPED, NULL);

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

fprintf(stderr, "Failed to open output file.\n");

CloseHandle(hSource);

return 1;

}

overlappedRead.hEvent = hEventRead;

overlappedWrite.hEvent = hEventWrite;

while (1) {

if (!ReadFile(hSource, currentBuffer, BUFFER\_SIZE, &bytesRead, &overlappedRead)) {

DWORD error = GetLastError();

if (error == ERROR\_HANDLE\_EOF) {

printf("End of file reached.\n");

break;

}

else if (error != ERROR\_IO\_PENDING) {

fprintf(stderr, "ReadFile failed with error: %lu\n", error);

break;

}

}

GetOverlappedResult(hSource, &overlappedRead, &bytesRead, TRUE);

transliterateBuffer(currentBuffer, processingBuffer, bytesRead);

if (!WriteFile(hDest, processingBuffer, bytesRead, &bytesWritten, &overlappedWrite)) {

if (GetLastError() != ERROR\_IO\_PENDING) {

fprintf(stderr, "WriteFile failed.\n");

break;

}

}

GetOverlappedResult(hDest, &overlappedWrite, &bytesWritten, TRUE);

overlappedRead.Offset += bytesRead;

overlappedWrite.Offset += bytesWritten;

}

CloseHandle(hSource);

CloseHandle(hDest);

CloseHandle(hEventRead);

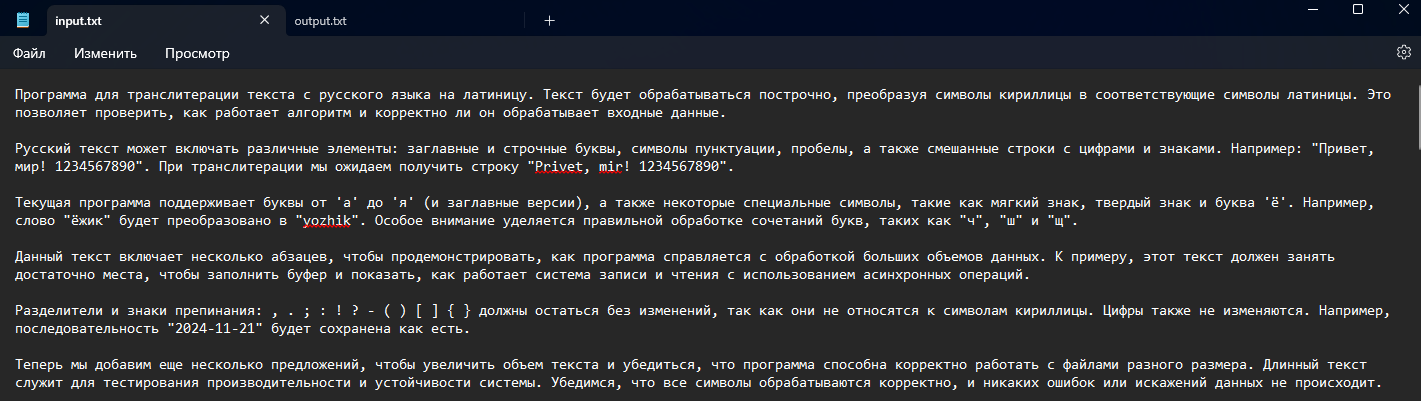
CloseHandle(hEventWrite);

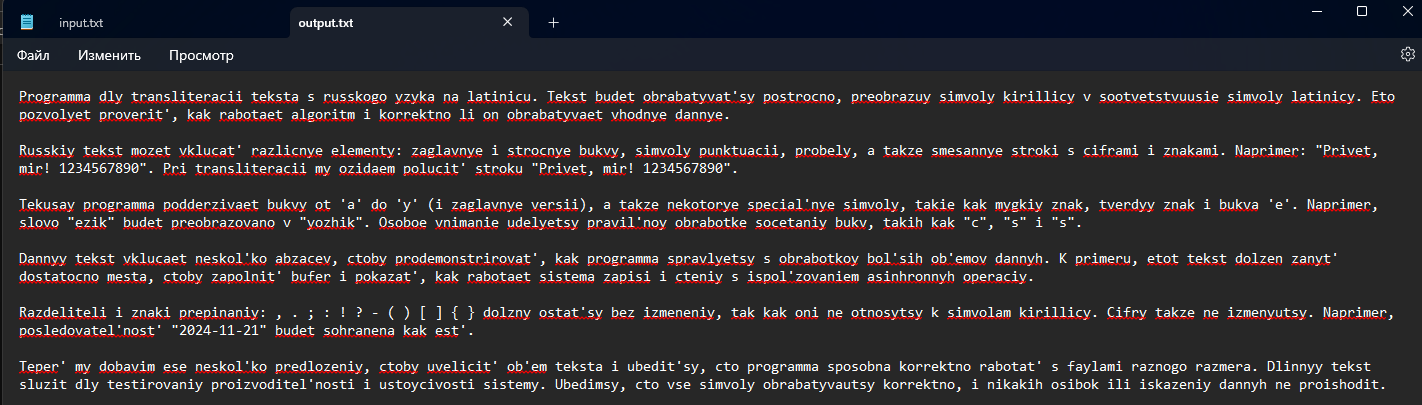
printf("Transliteration completed.\n");

return 0;

}

**Результат работы программы**





**Вывод**

Изучили способы работы с файлами в Win32 API. Изучили асинхронный режим работы с файлами.