#include <iostream>

#include <clocale>

#include <ctime>

#include "ErrorMsgText.h"

#include "Windows.h"

#define NAME L[\\\\Across\\pipe\\Tube](file:///\\\\Across\\pipe\\Tube) - определение константы NAME с именем и местоположением именованного канала.

#define STOP "STOP" - определение константы STOP для завершения обмена сообщениями по каналу.

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

HANDLE cH; - объявление дескриптора клиентского конца именованного канала.

DWORD lp; - объявление переменной для хранения количества переданных байт.

char buf[50]; - объявление буфера для чтения/записи данных по именованному каналу.

try {

cout << "ClientNP\n\n";

if ((cH = CreateFile(NAME, GENERIC\_READ | GENERIC\_WRITE,

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

NULL, OPEN\_EXISTING, NULL, NULL)) == INVALID\_HANDLE\_VALUE) {

throw SetPipeError("CreateFile: ", GetLastError());

} - создание клиентского конца именованного канала и его открытие для чтения/записи данных.

int countMessage; - объявление переменной для хранения количества сообщений.

cout << "Number of messages: "; - вывод сообщения для ввода количества сообщений.

cin >> countMessage; - ввод количества сообщений.

for (int i = 1; i <= countMessage; i++) { - цикл для отправки countMessage сообщений.

string obuf = "Hello from ClientNP " + to\_string(i); - формирование сообщения, которое будет отправлено по именованному каналу.

if (!WriteFile(cH, obuf.c\_str(), sizeof(obuf), &lp, NULL)) { - отправка сообщения по именованному каналу.

throw SetPipeError("WriteFile: ", GetLastError()); - генерация исключения в случае ошибки при отправке сообщения по именованному каналу.

}

if (!ReadFile(cH, buf, sizeof(buf), &lp, NULL)) { - чтение ответного сообщения с серверного конца именованного канала.

throw SetPipeError("ReadFile: ", GetLastError()); - генерация исключения в случае ошибки при чтении сообщения по именованному каналу.

}

cout << buf << endl; - вывод на экран ответного сообщения.

}

if (!WriteFile(cH, STOP, sizeof(STOP), &lp, NULL)) {

throw SetPipeError("WriteFile: ", GetLastError());

} происходит отправка сообщения "STOP" серверу с помощью функции WriteFile. Это сообщение сигнализирует серверу, что клиент закончил передачу сообщений и сервер может завершить свою работу.

if (!CloseHandle(cH)) { помощью функции CloseHandle происходит закрытие дескриптора канала связи, что освобождает занятые им ресурсы и позволяет другим процессам использовать данный канал связи. Если при закрытии дескриптора происходит ошибка, то генерируется исключение и выводится сообщение об ошибке.

throw SetPipeError("CloseHandle: ", GetLastError());

}

system("pause");

}

catch (string ErrorPipeText) {

cout << endl << ErrorPipeText;

}

}