Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №4

Дисциплина: «Операционные системы»

Тема «Межпроцессное взаимодействие на основе каналов и сокетов в операционной системе Windows»

Выполнил: Белов А.А.

Группа: 6205-010302D

Самара, 2023

**Задание на лабораторную работу**

Вариант 5

1. Задания выполняются на С++ в операционной системе Windows. В качестве дополнительного задания можно избрать любой другой язык программирования и другую операционную систему.
2. Число клиентов зависит от задания.
3. Реализация через именованные каналы

Сервер должен получать от одного единственного клиента сообщения в формате Арифметическая Операция Целое или Дробное Число (например, +322, -55, /400.54, \*322 и т.д.). После каждого такого ввода на сторону клиента должно быть передано сообщение со всей формулой на данный момент. Полученные от клиента части выражения нужно приписывать справа от уже имеющегося. Следует предусмотреть набор служебных команд для проверки выражения на корректность (в частности, деление на 0), очистка последовательности, расчет выражения. Следует также учесть при расчете приоритетность арифметических операций. 52 Так же должны быть предусмотрены такие вещи как набор служебных команд для очищения последовательности вычисления, расчета выражения, проверка корректности выражения (отсутствие в нём /0). Выражение должно считаться по правилам арифметики с соответствующим приоритетом операций.

**Текст программы**

Клиентская часть:

#include <iostream>

#include <Windows.h>

#include <string>

//#include <locale>

using namespace std;

string correct\_input() {

string buffer;// области памяти, где будут храниться считанные или записываемые данные.

string res;

bool flag = true;

while (flag){

cout << "Enter the data to send to the server: ";

getline(cin, buffer);

if (buffer == "/exit" || buffer == "=" || buffer == "/clean") {

return buffer;

}

string digit = buffer.substr(1);

if (buffer[0] == '+' || buffer[0] == '-' || buffer[0] == '\*' || (buffer[0] == '/' && digit != "0")) {

try {

if (stod(digit)) {

cout << "Success" << endl;

res += buffer[0];

double ss = stod(digit);

res += to\_string(ss);

flag = false;

}

}

catch (invalid\_argument e) {

cout << "Error, input digit: " << endl;

}

}

else {

cout << "Error! Write a sign at the beginning " << endl;

}

}

//cout << res << endl << endl;

return res;

}

int main() {

//setlocale(LC\_ALL, "Rus");

HANDLE hPipe; // переменная для хранения дескриптора именного канала

char buffer[1024]; // области памяти, где будут храниться считанные или записываемые данные.

DWORD dwWritten; // количество байт, успешно записанных в именной канал при использовании функции WriteFile

DWORD dwRead;

LPCWSTR pipeName = L"\\\\.\\pipe\\MyNamedPipe";

string calc;

// Подключение к именному каналу

hPipe = CreateFile(

pipeName, // Имя канала

GENERIC\_READ | GENERIC\_WRITE, // Режим доступа

0, // Нет совместного использования

NULL, // Атрибуты защиты

OPEN\_EXISTING, // Открытие существующего канала

0, // Атрибуты файла

NULL); // Хэндл шаблона файла)

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

cout << "Connection error to the named channel: " << GetLastError() << endl;

return 1;

}

while (true) {

Sleep(1);

Sleep(5);

//buffer[1024] = correct\_input();

string num = correct\_input();

char arr[sizeof(num)];

//cout << sizeof(num);

// Преобразование std::string в char[]

strcpy\_s(arr, num.c\_str());

// Отправка данных в канал

if (WriteFile(hPipe, arr, sizeof(arr), &dwWritten, NULL)) {

//Sleep(5);

if (dwWritten > 0)

cout << "Sent to server " << dwWritten << " byte: " << arr << endl;

else

cout << "Error send data: " << GetLastError() << endl;

if (num == "/exit") {

cout << ":) " << endl;

CloseHandle(hPipe);

return 0;

}

if (ReadFile(hPipe, buffer, sizeof(buffer), &dwRead, NULL)) {

//Sleep(5);

char mas[1024];

size\_t length = strnlen(buffer, sizeof(mas)); // определение длины буфера до '\0'

strncpy\_s(mas, buffer, length); // копирование символов из buffer в arr до '\0'

mas[length] = '\0'; // добавление нулевого символа в конец arr, чтобы получить корректную строку

cout << "Received from cerver " << dwRead << " byte: " << mas << endl;

// Закрытие канала

//CloseHandle(hPipe);

}

}

}

// Закрытие канала

CloseHandle(hPipe);

return 0;

}

Серверная часть:

#include <iostream>

#include <Windows.h>

#include <string>

using namespace std;

int main() {

//setlocale(LC\_ALL, "Rus"); //он нахуй всё ломает блять

HANDLE hPipe;

char buffer[1024];

DWORD dwRead;

DWORD dwWritten; // количество байт, успешно записанных в именной канал при использовании функции WriteFile

LPCWSTR pipeName = L"\\\\.\\pipe\\MyNamedPipe";

string calc = "";

double res = 0;

// Создание именного канала

hPipe = CreateNamedPipe(

pipeName, // Имя канала

PIPE\_ACCESS\_DUPLEX, // Режим доступа к каналу

PIPE\_TYPE\_MESSAGE | // Тип канала

PIPE\_READMODE\_MESSAGE | // Режим чтения

PIPE\_WAIT, // Режим ожидания

PIPE\_UNLIMITED\_INSTANCES, // Максимальное количество экземпляров

sizeof(buffer), // Размер буфера выходного буфера

sizeof(buffer), // Размер буфера входного буфера

0, // Время ожидания операций

NULL); // Атрибуты защиты

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

cout << "Error creating a named channel: " << GetLastError() << endl;

return 1;

}

cout << "Waiting for client connection..." << endl;

if (ConnectNamedPipe(hPipe, NULL)) {

cout << "The client is connected." << endl;

while (true) {

//Sleep(3);

// Чтение данных из канала

if (ReadFile(hPipe, buffer, sizeof(buffer), &dwRead, NULL)) {

bool flag = true;

//cout << "Получено " << dwRead << " байт: " << buffer << endl;

char arr[1024];

size\_t length = strnlen(buffer, sizeof(arr)); // определение длины буфера до '\0'

strncpy\_s(arr, buffer, length); // копирование символов из buffer в arr до '\0'

arr[length] = '\0'; // добавление нулевого символа в конец arr, чтобы получить корректную строку

string befor(arr, length);

if (befor == "/exit") {

cout << ":)" << endl;

CloseHandle(hPipe);

return 0;

}

if (befor == "/clean") {

calc = "";

res = 0;

memset(buffer, 0, sizeof(buffer));

memset(arr, 0, sizeof(arr));

WriteFile(hPipe, arr, sizeof(arr), &dwWritten, NULL);

flag = false;

}

if (flag) {

bool flag\_2 = true;

calc.append(befor);

if (befor[0] == '+' || befor[0] == '-' || befor[0] == '\*' || befor[0] == '/') {

string digit = befor.substr(1);

double num = stod(digit);

if (befor[0] == '+')

res += num;

if (befor[0] == '-')

res -= num;

if (befor[0] == '\*')

res \*= num;

if (befor[0] == '/')

res /= num;

}

if (befor[0] == '=') {

calc.append(to\_string(res));

strcpy\_s(arr, calc.c\_str());

WriteFile(hPipe, arr, sizeof(arr), &dwWritten, NULL);

cout << "Result: " << res << endl;

cout << "Write /clear to clear area " << endl;

flag\_2 = false;

}

cout << "Received " << dwRead << " byte: " << calc << endl;

if (flag\_2) {

strcpy\_s(arr, calc.c\_str());

if (WriteFile(hPipe, arr, sizeof(arr), &dwWritten, NULL)) {

//Sleep(5);

if (dwWritten > 0)

cout << "Send to client " << dwWritten << " byte: " << arr << endl;

else

cout << "Error send data: " << GetLastError() << endl;

}

//Sleep(5);

}

}

}

}

// Закрытие канала

CloseHandle(hPipe);

}

return 0;

}