Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ»

Институт информационных и вычислительных технологий

Кафедра Управления и интеллектуальных технологий

**Отчёт по лабораторной работе №3**

**По курсу «Системное программное обеспечение»**

**Межпроцессное взаимодействие**

Выполнил: студенты группы А-01-19

Поташов С.Е.

Проверил:

Козлюк Д.А.

Вариант 2

# Москва 2022

# Задание на лабораторную работу: Написать пару программ, взаимодействующих через именованные каналы. Программа-сервер хранит строки-значения по строкам-ключам. Программа-клиент, отправляя программе-серверу команды, добавляет, удаляет и получает значения.

## Сервер:

**Задача сервера - создать именованный канал, дождаться подключения процесса-клиента, обработать запросы клиента и выдать необходимую информацию.**

### Создание именованного канал

map<string, string> data{};

cout<< "Enter pipe name: ";

string name\_pipe;

getline(cin,name\_pipe);

//the named pipe name

auto path = "\\\\.\\pipe\\" + name\_pipe;

//create named pipe

auto pipe = CreateNamedPipeA(path.c\_str(),

PIPE\_ACCESS\_DUPLEX,

PIPE\_TYPE\_MESSAGE,

PIPE\_UNLIMITED\_INSTANCES,

BUFFER\_SIZE,

BUFFER\_SIZE,

0,

NULL);

if (pipe == INVALID\_HANDLE\_VALUE)

{

cout << "Error:"<< GetLastError()<<'\n';

return 0;

}

# 

# Основной цикл работы программы:

bool flag = true;

while (flag)

{

//Waiting for a connection

std::cout << "The client connecting, wait please"<<'\n';

if (ConnectNamedPipe(pipe, NULL))

{

std::cout << "connection successful"<<'\n';

}

else

{

std::cout << "Error:"<< GetLastError()<<'\n';

CloseHandle(pipe);

}

//ReadFile and commands

string command (BUFFER\_SIZE, '\0');

string cmd,key,value, response{};

while(true)

{

std::cout << "Waiting for a command..."<<'\n';

if (!ReadFile(pipe,&command[0], command.size(), NULL, NULL))

{

std::cout << "Error:" << GetLastError()<<'\n';

}

command.resize(command.find('\0'));

std::istringstream parser{command};

parser >> cmd;

cout << "Command: " << command<< '\n';

if(cmd == "set")

set\_f(pipe, data, &parser, key, value);

else if (cmd == "get")

get\_f(pipe, data, &parser);

else if (cmd == "list")

list\_f(pipe, data);

else if (cmd == "delete")

delete\_f(pipe, data, &parser);

else if (cmd == "quit")

{

DisconnectNamedPipe(pipe);

break;

}

else

{

cout << "Non-existent command! Try again: "<<'\n';

}

command.replace(0, command.size(), command.size(), '\0');

command.resize(BUFFER\_SIZE, '\0');

//response.clear();

name\_pipe.clear();

value.clear();

}

//Continue working or end session

char proceed;

std::cout << "Do you want to continue?[y or n]: "<<'\n';

while (true)

{

std::cin >> proceed;

if (proceed == 'y')

{

break;

}

else if (proceed == 'n')

{

CloseHandle(pipe);

flag = false;

break;

}

else

{

std::cout << "Non-existent command! Try again: "<<'\n';

continue;

}

}

}

# Функция сохранение значения по ключу. Формат команды: set ключ значение необходимо сохранить в памяти значение под указанным ключом и записать в канал строку acknowledged.

int set\_f (HANDLE pipe, map<string, string> &data, istringstream \*parser, string &key, string &value)

{

\*parser >> key >> value;

data[key] = value;

string response = "acknowledged";

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

return 1;

# }

# Функция получения значения по ключу. Формат команды: get ключ Если ключ имеется в хранилище, следует записать в канал строку в формате found значение В противном случает следует записать в канал строку missing

int get\_f (HANDLE pipe, map<string, string> data, istringstream \*parser)

{

string key;

\*parser >> key;

if (data.find(key) != data.end())

{

std::string response = "found " + data[key];

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

}

else

{

std::string response = "missing";

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

}

}

# Функция получения списка ключей в хранилище. Формат команды: list Необходимо записать в канал строку, содержащую через пробел все имеющиеся в хранилище ключи.

int list\_f(HANDLE pipe, map<string, string> data)

{

string response;

for (auto& i : data)

response += i.first + " ";

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

}

# Функция удаления значения под заданным ключом. Формат команды: delete ключ Если ключ присутствует в хранилище, следует записать в канал строку deleted, иначе — строку missing

int delete\_f(HANDLE pipe, map<string, string> &data, istringstream \*parser)

{

string key;

string response;

\*parser >> key;

auto del = data.find(key);

if (del != data.end())

{

data.erase(del);

response = "deleted";

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

}

else

{

response = "missing";

WriteFile(pipe, response.c\_str(), response.size(), NULL, NULL);

}

}

# Клиент:

# Подключение к именованному каналу:

//connecting to the named pipe by client

string name\_pipe;

cout<<"Enter name pipe:";

getline(cin >> ws, name\_pipe);

auto path = "\\\\.\\pipe\\" + name\_pipe;

auto client\_pipe = CreateFile(path.c\_str(),

GENERIC\_READ | GENERIC\_WRITE,

0,

NULL,

OPEN\_EXISTING,

FILE\_ATTRIBUTE\_NORMAL,

NULL);

if (client\_pipe == INVALID\_HANDLE\_VALUE)

{

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

return 0;

}

string command (BUFFER\_SIZE,'\0');

Основной цикл работы:

//sending a request to the server

while (true)

{

cout << ">> ";

getline(cin >> ws, command);

WriteFile(client\_pipe, command.c\_str(), command.size(), NULL, NULL);

if (command == "quit")

{

CloseHandle(client\_pipe);

break;

}

else if (command.substr(0, 3) == "set" || command.substr(0, 3) == "get" ||

command == "list" || command.substr(0, 6) == "delete")

{

string response(64,'\0');

ReadFile(client\_pipe, &response[0], response.size(), NULL, NULL);

cout<<response<<endl;

}

else

{

cout << "Non-existent command! Try again: "<<'\n';

continue;

}

}

# Пример работы программы:

# Со стороны клиента:

# 

# Со стороны сервера:

# 