# Лабораторная работа № 2

по курсу Операционные системы:

Процессы и взаимодействие между ними

Выполнил студент группы М80-204Б МАИ Дюсекеев Алишер

Оценка_____

## Цель работы

Приобретение практических навыков в:

- Управление процессами в ОС
- Обеспечение обмена данных между процессами посредством каналов

## Задание

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решения задачи один или несколько дочерних процессов.
Взаимодействие между процессами осуществляется через системные сигналы/события и/или каналы (pipe).
Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

## Варианты задания

**Рекурсивное вычисление чисел Фибоначчи, где каждый отдельный уровень рекурсии вычисляется в отдельном процессе.**

# Содержание OsLab2.cpp

```
#include "stdafx.h"
#include <Windows.h>

int RecursFibon(int cur, int prev, int numFib, int counter) {
	counter++;
	if (numFib == 1) return 0;
	if (numFib == 2) return 1;
	if (numFib == counter) return cur;

	SECURITY_ATTRIBUTES sa;
	sa.nLength = sizeof(SECURITY_ATTRIBUTES);
	sa.lpSecurityDescriptor = NULL;
	sa.bInheritHandle = TRUE;
	DWORD writeBytes, readBytes;

	PROCESS_INFORMATION ProcessInfo; //This is what we get as an [out] parameter
	ZeroMemory(&ProcessInfo, sizeof(PROCESS_INFORMATION));
	STARTUPINFO StartupInfo; //This is an [in] parameter
	TCHAR lpszClientPath[] = L"OsLab2client";
	ZeroMemory(&StartupInfo, sizeof(StartupInfo));
	StartupInfo.cb = sizeof(STARTUPINFO); //Only compulsory field
	HANDLE pipe1Read, pipe1Write, pipe2Read, pipe2Write;
	CreatePipe(&pipe1Read, &pipe1Write, &sa, 0);
	CreatePipe(&pipe2Read, &pipe2Write, &sa, 0);
	StartupInfo.dwFlags = STARTF_USESTDHANDLES;
	StartupInfo.hStdInput = pipe1Read;
	StartupInfo.hStdOutput = pipe2Write;

	bool process = CreateProcess(NULL,
		lpszClientPath,
		NULL, NULL, true,
		CREATE_NO_WINDOW,// CREATE_NEW_CONSOLE|CREATE_SUSPENDED
		NULL, NULL,
		&StartupInfo,
		&ProcessInfo);
	CloseHandle(pipe1Read);
	CloseHandle(pipe2Write);

	WriteFile(pipe1Write, &cur, sizeof(int), &writeBytes, NULL);
	WriteFile(pipe1Write, &prev, sizeof(int), &writeBytes, NULL);
	ReadFile(pipe2Read, &cur, sizeof(int), &readBytes, NULL);
	ReadFile(pipe2Read, &prev, sizeof(int), &readBytes, NULL);

	CloseHandle(pipe2Read);
	CloseHandle(pipe1Write);
	CloseHandle(ProcessInfo.hThread);
	CloseHandle(ProcessInfo.hProcess);

	cur = RecursFibon(cur, prev, numFib, counter);
	return cur;
}

int _tmain(int argc, _TCHAR* argv[])
{

	int cur = 1;
	int prev = 0;
	int numFib = 20;
	int res = 0;
	res = RecursFibon(cur, prev, numFib, 1);


	printf("%d\n", res);
	system("pause");
	return 0;
}
```

## Содержание OsLab2client.cpp

```
#include "stdafx.h"
#include <Windows.h>

int _tmain(int argc, _TCHAR* argv[])
{
	/*DWORD mode;
	bool re = GetConsoleMode(GetStdHandle(STD_INPUT_HANDLE), &mode);
	bool res = SetConsoleMode(GetStdHandle(STD_INPUT_HANDLE), 0);
	GetConsoleMode(GetStdHandle(STD_INPUT_HANDLE), &mode);*/
	HANDLE inH = GetStdHandle(STD_INPUT_HANDLE);
	HANDLE outH = GetStdHandle(STD_OUTPUT_HANDLE);
	int cur;
	int prev;
	int tmp;
	DWORD readBytes, writeBytes;
	if (!ReadFile(inH, &cur, sizeof(int), &readBytes, NULL))
		return -1;
	if (!ReadFile(inH, &prev, sizeof(int), &readBytes, NULL))
		return -3;
	tmp = cur;
	cur += prev;
	prev = tmp;

	if (!WriteFile(outH, &cur, sizeof(int), &writeBytes, NULL))
		return -2;
	if (!WriteFile(outH, &prev, sizeof(int), &writeBytes, NULL))
		return -4;
	return 0;
}
```

## Вывод программы

```
4181

Для продолжения нажмите любую клавишу . . .
```

## Суть работы программы

Программа запускает рекурсивную функцию, в которой происходит создание процесса по вычислению следующего числа Фибоначчи. Передача предыдущего числа и нынешнего числа производится через Pipe (CreatePipe). В обратную сторону передается результат работы: следующее число Фибонначи и нынешнее число. При нахождении нужного числа рекурсия развертывается и возвращает результат.

## Системные вызовы использованные в лабораторной

- BOOL WINAPI CreateProcess(...) - создание нового процесса
- WaitForSingleObject(...) - ожидание завершения процесса
- ExitProcess(...) - завершение выполнения процесса
- CreateFile/SetNamedPipeHandleState - создание именованного канала и установления режима его использования
- OpenFile(...) - открытие нового файла
- CreatePipe(...) - создание безымянного канала
- CreateFile(...) - создание нового файла
- CloseHandle(...) - закрытие объекта ОС по "заголовку". Подходит для закрытия файлов.

## Вывод

Общение между процессами позволяет разделять нагрузку и распараллеливать работу программы. Каналы или pipes это очень удобное средство для передачи информации между процессами. Через pipes можно отправлять на обработку как базовые элементы, так и целые структуры данных. И программа, из которой посылается пакет данных, и процесс, принимающий эти данные, должны ждать ответа друг от друга. В случае неуспеха данные не обработаются правильно, и программа выведет неверный ответ. В данной лабораторной был создан именованный канал для работы в дуплексном режиме. Данные должны передаваться и справа-налево, и слева-направо. Программа работает и завершается успешно.