БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий

и робототехники

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

О Т Ч Ё Т

о лабораторной работе № 5

Дисциплина

«Системное программирование»

Тема

«Процессы (часть 1)»

Выполнил: студент гр. 10702217 Храмков Д. С.

Проверил: Разорёнов Н. А.

Минск 2019

***Лабораторная работа № 5***

**ПРОЦЕССЫ (часть 1)**

**Цель работы:** ознакомление с основами создания и управления процессами в ОС WINDOWS.

**Изучаемые вопросы**

1. Виды процессов.
2. Структуры STARTUPINFO, PROCESS\_INFORMATION.
3. Создание процесса.
4. Класс приоритета процесса.
5. Идентификатор процесса.
6. Дочерние процессы.
7. Наследование дочерними процессами ресурсов (файлов) родительского процесса.
8. Функции Win32 для управления процессами.
9. Окончание процесса.

***Постановка задачи***

1. Разработать приложение, состоящее из трех процессов (головного и двух дочерних). В головном процессе создаются/открываются два файла: текстовый (несколько строк) и двоичный (ряд арифметических данных). Информация об открытых файлах (дескрипторы) передается в дочерние процессы на этапе их создания через командную строку.
2. В главном процесса запускаются два дочерних процесса, которые:

* первый – дописывает строку/строки в текстовый файл;
* второй – дописывает число/числа в бинарный файл;
* процессы отображают информацию файлов до и после их модификации.

1. После возвращения из дочерних процессов главный процесс отображает:

* Содержимое модифицированных файлов и закрывает их;
* Системную информацию о процессе (GetProcessInformation).

1. **Виды процессов.**

**Существуют следующие виды процессов:**

* **Пользовательские;**
* **Системные;**
* **Дочерние;**
* **Родительские.**

1. **Структуры STARTUPINFO, PROCESS\_INFORMATION.**

**Структура STARTUPINFO используется функцией** CreateProcess **для определения свойств главного окна, если новое окно создается для нового процесса. Для создания достаточно указать размер структуры.**

**Структура PROCESS\_INFORMATION нужна для хранения информации о созданном процессе и его главном потоке.**

STARTUPINFO si = {

sizeof(si)

};

PROCESS\_INFORMATION pi[2];

CreateProcess(exeTxtFile, cmdTxtParam, NULL, NULL, TRUE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &si, &pi[0]);

1. **Создание процесса.**

**Процесс создается при помощи команды** CreateProcess(exeTxtFile, cmdTxtParam, NULL, NULL, TRUE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &si, &pi[0])**. При создании процесса можно задавать какую программу он будет выполнять, параметры этой программы, приоритет и переменные для хранения информации о процессе.**

1. **Класс приоритета процесса.**

**Существует 6 классов приоритета:**

* IDLE\_PRIORITY\_CLASS;
* BELOW\_NORMAL\_PRIORITY\_CLASS;
* NORMAL\_PRIORITY\_CLASS;
* ABOVE\_NORMAL\_PRIORITY\_CLASS;
* HIGH\_PRIORITY\_CLASS;
* REALTIME\_PRIORITY\_CLASS.

Чтобы присвоить один из вышеперечисленных приоритетов процессу, надо указать его в параметре fdwCreate функции CreateProcess().

CreateProcess(exeBinFile, cmdBinParam, NULL, NULL, TRUE, ABOVE\_NORMAL\_PRIORITY\_CLASS, NULL, NULL, &si, &pi[1]);

1. **Идентификатор процесса.**

**Идентификатор процесса – уникальный номер, характеризующий данный процесс. Хранится в переменной структуры PROCESS\_INFORMATION.**

pi[1].dwProcessId;

1. **Дочерние процессы.**

**Процессы, которые создаются функцией CreateProcess() в родительском – называются дочерними. Дочерние процессы наследуют атрибуты родительского процесса.**

CreateProcess(exeTxtFile, cmdTxtParam, NULL, NULL, TRUE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &si, &pi[0]);

1. **Наследование дочерними процессами ресурсов (файлов) родительского процесса.**

**Для передачи в дописывающий процесс дескриптора файла, нужно убедиться, что последний имеет указатель на структуру SECURITY\_ATTRIBUTES. Затем передать описатель файла через параметры командной строки. Для получения дочерним процессом описателя используется команда GetCommandLine().**

SECURITY\_ATTRIBUTES sa;

sa.nLength = sizeof(SECURITY\_ATTRIBUTES);

sa.lpSecurityDescriptor = NULL;

sa.bInheritHandle = TRUE;

hTxtFile = CreateFile(txtFile, GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ, &sa, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL)) == INVALID\_HANDLE\_VALUE);

wsprintf(cmdTxtParam, L"%d", hTxtFile);

CreateProcess(exeTxtFile, cmdTxtParam, NULL, NULL, TRUE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &si, &pi[0]);

if (hFile = (HANDLE)\_wtoi(GetCommandLineW())) {

…

}

1. **Функции Win32 для управления процессами.**

**Для управления процессами существует ряд API-функций:**

***CreateProcess*(NULL, str, NULL, NULL, true, NULL, NULL ,**

**NULL, &si, &pi) – функция создает новый процесс;**

***WaitForSingleObject*(pi.hProcess, INFINITE) – функция приоста-навливает выполнение потока родительского процесса, пока не за-вершится дочерний процесс pi.hProcess;**

***CloseHandle*(hFileText) – уменьшает значения счетчиков объек-тов на единицу. Когда счетчик уменьшится до нуля, освобождается память, занимаемая этим объектом;**

***PTSTR* *GetCommandLine*() – получает указатель на полную ко-мандную строку;**

***DWORD* *GetEnvironmentVariable*(PCTSTR pszName, PTSTR pszValue, DWORD cchValue) – позволяет выявлять присутствие той или иной переменной окружения и определять ее значение;**

***DWORD GetCurrentDirectory*(DWORD cchCurDir, PTSTR pszCurDir); – получает текущий каталог и диск для процесса;**

***BOOL SetCurrentDirectory*(PCTSTR pszCurDir) – устанавливает текущий каталог и диск для процесса;**

***HANDLE OpenProcess*(DWORD fdwAccess, BOOL fInherit,**

**DWORD, IDProcess) – возвращает дескриптор существующего в системе процесса.**

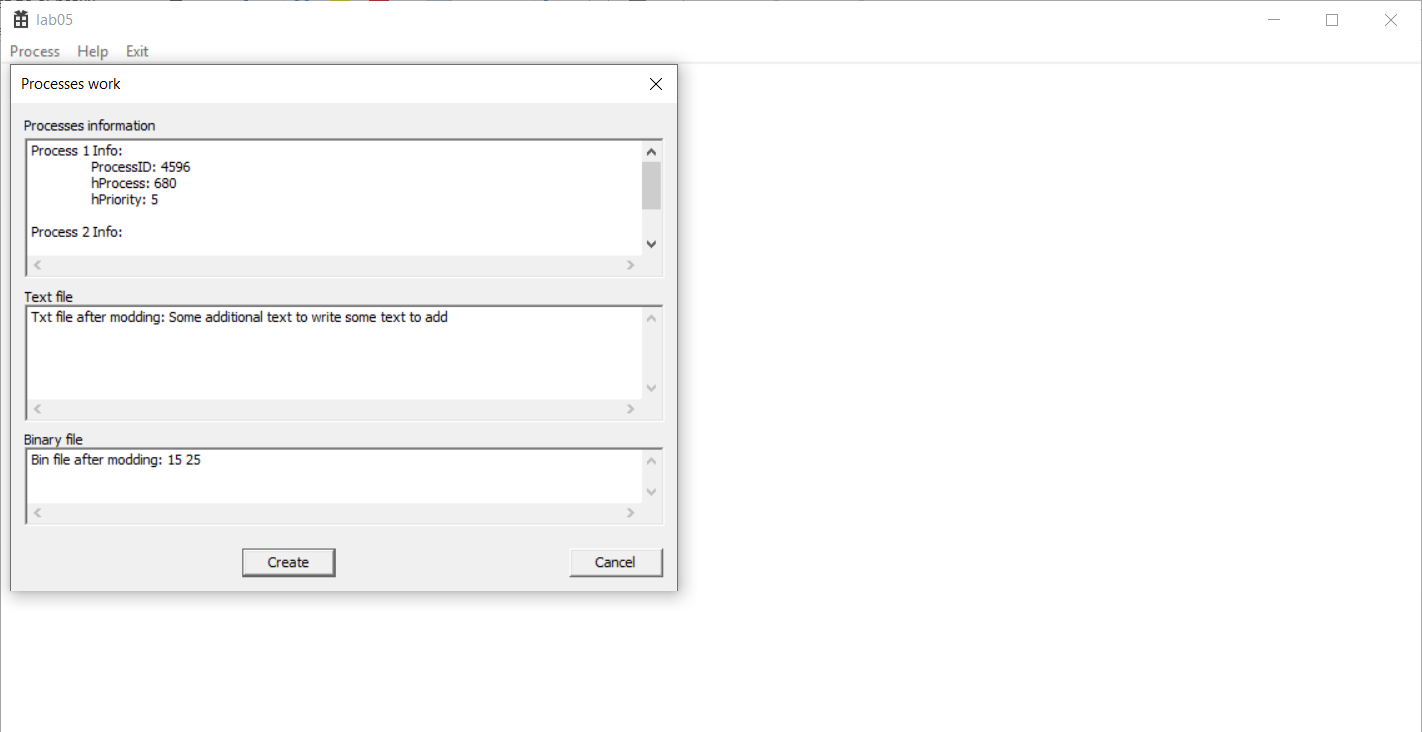
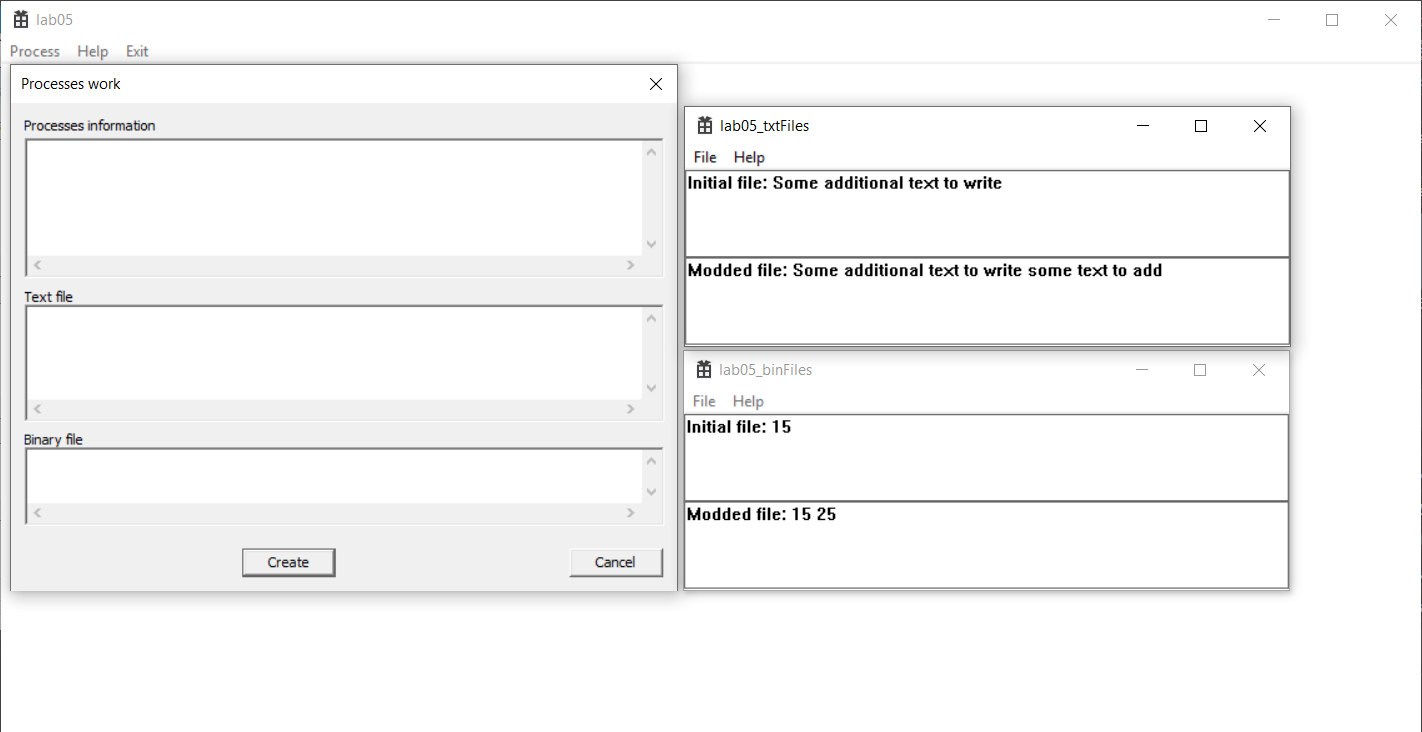
WaitForSingleObject(pi[0].hProcess, INFINITE);

1. **Окончание процесса.**

**Процесс завершается если:**

1. **Входная функция первичного потока возвратила управление;**
2. **Один из потоков процесса вызвал функцию ExitProcess;**
3. **Поток другого процесса вызвал функцию TerminateProcess;**
4. **Все потоки процесса умерли по своей воле.**

**Результаты работы программы:**



**Выводы:**

1. **Научились создавать процессы;**
2. **Научились передавать дескриптор файла как параметр командной строки;**
3. **Научились задавать приоритет процессов;**
4. **Научились получать информацию о процесса.**