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 № 7

Дисциплина

«Системное программирование»

Тема

«Процессы (часть 2)»

Выполнил: студент гр. 10702217 Храмков Д. С.

Проверил: Разорёнов Н. А.

Минск 2019

***Лабораторная работа № 7***

**ПРОЦЕССЫ (часть 2)**

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

**Изучаемые вопросы**

1. Переменные окружения процесса: структура, значения.
2. Передача информации между процессами через среду процесса.
3. Список процессов.
4. Кто родитель процесса.
5. Текущая рабочая директория процесса.
6. Время выполнения процесса.
7. Наследование дочерними процессами среды родительского процесса.

***Постановка задачи***

1. Модифицировать программу лабораторной работы № 5 так, чтобы она выводила в окно информацию по изучаемым вопросам. Приложение может состоять из трех процессов (головного и двух дочерних). В родительском процессе создается и выводится список процессов, который обновляется через 3 с, и другая информация по процессу.
2. В главном процессе запускаются два дочерних процесса, где первый выводит PID, среду и PID родителя, а второй дописывает число в бинарный файл. Первый дочерний процесс записывает отображаемую информацию в текстовый файл.
3. После возвращения из дочерних процессов главный процесс отображает содержимое файла.
4. **Переменные окружения процесса: структура, значения.**

**В создание любого процесса вовлечён блок переменных окружения, или среда. Может использоваться для более тонкой настройки приложений.**

LPVOID lpvEnv = GetEnvironmentStrings();

LPCTSTR lpszVariable = (LPCTSTR)lpvEnv;

while (lpszVariable[0] != '\0') {

int size = \_tcslen(lpszVariable);

lpszVariable += \_tcslen(lpszVariable) + 1;

}

FreeEnvironmentStrings((LPTSTR)lpvEnv);

1. **Передача информации между процессами через среду процесса.**

**Чтобы задать переменную среды процесса нужно выполнить функцию** SetEnvironmentVariable() **Для получения переменной, переданной из родительского процесса дочернему, используется команда** GetEnvironmentVariable()**.**

DWORD procID = GetCurrentProcessId();

WCHAR parentProcID[20] = L"";

wsprintf(parentProcID, L"%d", procID);

SetEnvironmentVariable(L"parentProcID", parentProcID);

WCHAR parentProcID[20];

GetEnvironmentVariable(L"parentProcID", parentProcID, 20);

1. **Список процессов.**

**Список процессов представляет из себя снимок состояния системных объектов в определённый момент времени. Для получения такового пользуемся** CreateToolhelp32Snapshot()**;**

HANDLE hSnap = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

PROCESSENTRY32 pEntry;

pEntry.dwSize = sizeof(PROCESSENTRY32);

wstring message = L"All processes information: \r\n";

if (Process32First(hSnap, &pEntry)) {

do {

message += L"\t" + to\_wstring(pEntry.th32ProcessID) + L" " + to\_wstring(pEntry.th32ParentProcessID) + L" " + pEntry.szExeFile + wstring(L"\r\n");

} while (Process32Next(hSnap, &pEntry));

}

1. **Кто родитель процесса.**

**Родителя процесса можно получить из списка процессов, посмотрев ID родителя интересующего процесса, а затем найти процесс с этим ID.**

HANDLE hSnap = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

1. **Текущая рабочая директория процесса.**

**Текущей называется директория процесса, в которой он выполняет операции над файлами.**

TCHAR szDir[255];

GetCurrentDirectory(255, szDir);

1. **Время выполнения процесса.**

**Для того, чтобы процесс выполнялся корректно, он должен затратить некоторое время на загрузку ресурсов. В Win32 API можно отобразить следующие времена: время создания, время ядра, время пользователя и т.д.**

FileTimeToSystemTime(&creationTime, &systemTime);

time += L"\tCreation time: " + to\_wstring(systemTime.wHour) + L":" + to\_wstring(systemTime.wMinute) + L":" + to\_wstring(systemTime.wSecond) + L"\r\n";

FileTimeToSystemTime(&userTime, &systemTime);

time += L"\tUser time: " + to\_wstring(systemTime.wHour) + L":" + to\_wstring(systemTime.wMinute) + L":" + to\_wstring(systemTime.wSecond) + L"\r\n";

FileTimeToSystemTime(&kernelTime, &systemTime);

time += L"\tKernel time: " + to\_wstring(systemTime.wHour) + L":" + to\_wstring(systemTime.wMinute) + L":" + to\_wstring(systemTime.wSecond) + L"\r\n";

1. **Наследование дочерними процессами среды родительского процесса.**

**При создании процесса есть возможность передать ему среду родительского процесса, установив переменную со средой на место lpEnvironment.**

while (lpszVariable[0] != '\0') {

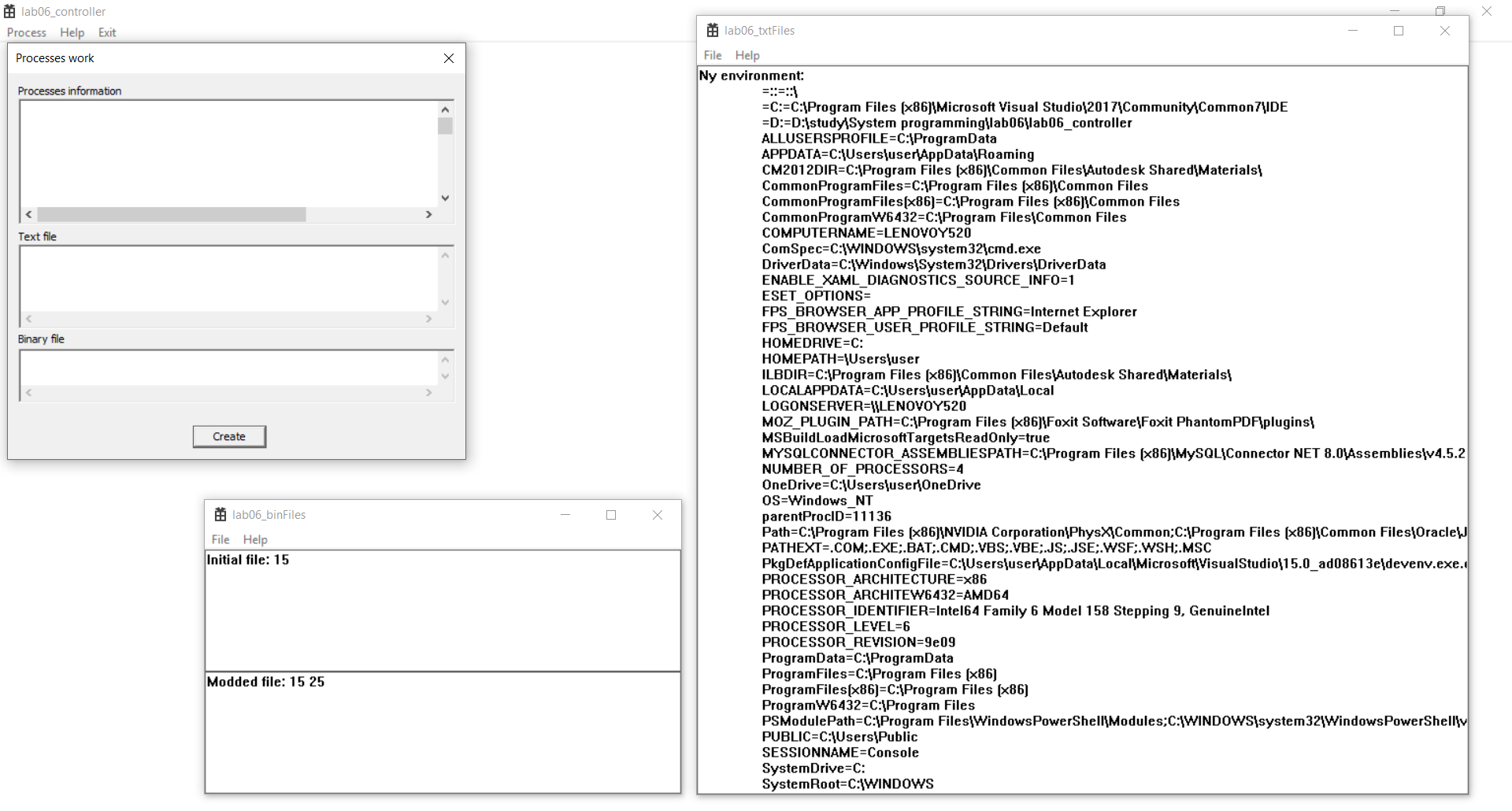
int size = \_tcslen(lpszVariable);

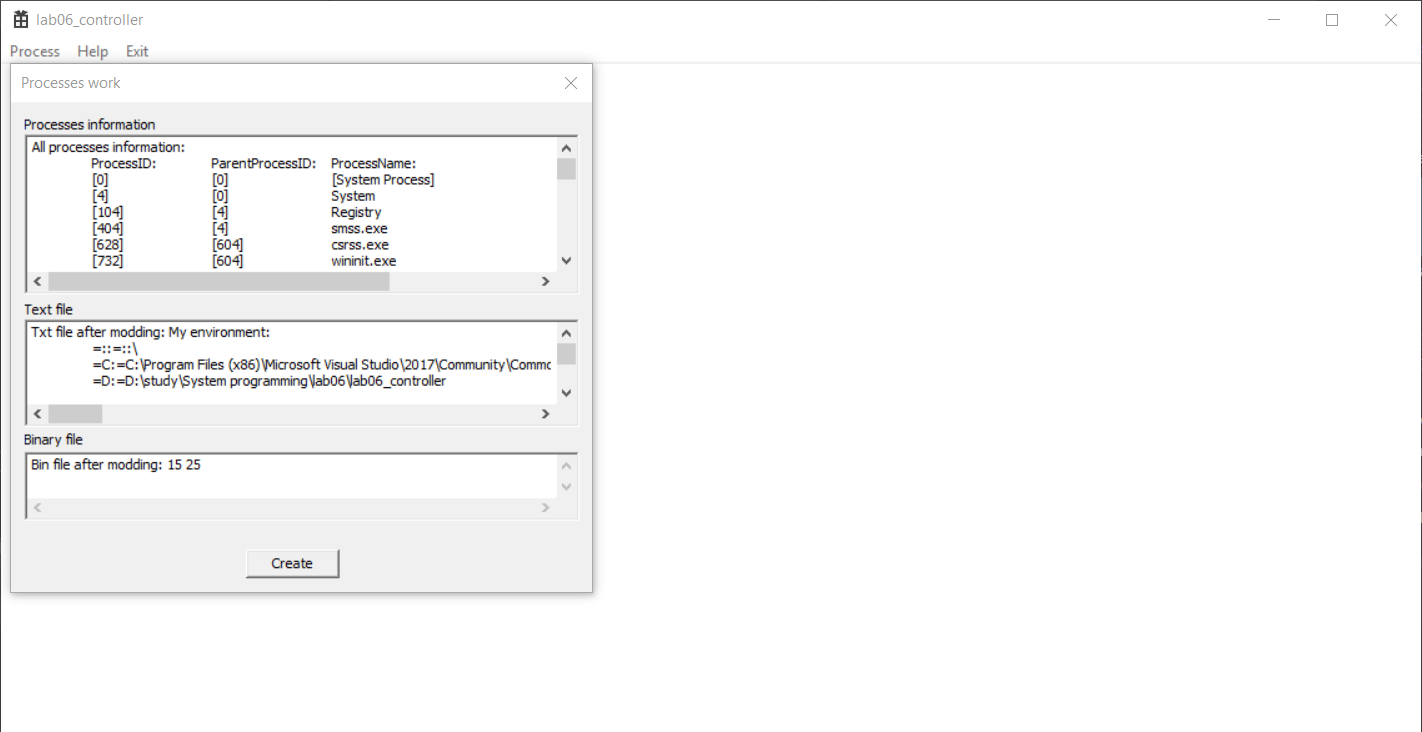
lpszVariable += \_tcslen(lpszVariable) + 1;

}

FreeEnvironmentStrings((LPTSTR)lpvEnv);

**Результаты работы программы:**





**Выводы:**

1. **Научились задавать переменные окружения процесса;**
2. **Научились выводить список процессов;**
3. **Научились определять родителя процесса;**
4. **Научились отслеживать время работы процесса.**