**Лабораторная работа №2**

**Управление процессами I**

**Цель работы:** Получение практических навыков в работе с процессами, а также с некоторыми ресурсами доступными процессам.

**Темы для предварительного изучения:**

* Утилита Sysinternals Process Explorer ([ссылка](https://learn.microsoft.com/en-us/sysinternals/downloads/process-explorer)).

**Постановка задачи для Windows:**

**Приложение Lab-02x:**

Программа выполняет простейший цикл с заданным количеством итераций, при чём каждая итерация выполняется с задержкой 500 мс. Программа должна выводить PID на каждой итерации и перед стартом выводить количество итераций. Количество итераций передается в программу через аргументы командной строки. Если не задан аргумент командной строки приложение завершает свою работу с ошибкой. (Использовать **ExitProcess**)

**Приложение Lab-02a:**

Программа создаёт три дочерних процесса с помощью **CreateProcess**. Все три процесса должны создаваться для программы **Lab-02x**, при чём для передачи имени исполняемого файла и параметра количества итераций должны быть выполнены следующие условия:

* При создании первого процесса используется только первый параметр функции **CreateProcess**.
* При создании второго процесса используется только второй параметр функции **CreateProcess**.
* При создании третьего процесса для передачи имени программы используется – первый параметр, для количества итераций – второй.

Родительский процесс должен дождаться завершения всех дочерних процессов (выполняются одновременно).

***Вопрос****: Все ли процессы созданы успешно? Если нет, почему?*

Также предусмотреть обработку ошибок и очистку ресурсов. Использовать Process Explorer для просмотра информации о процессах и проверки дескрипторов дочерних процессов.

**Модификация Lab-02х:**

Изменить Lab-02х так, чтобы количество итераций можно было получить через переменную окружения ITER\_NUM, если аргумент командной строки не задан. Если не заданы ни переменная окружения, ни аргумент командной строки приложение завершает свою работу с ошибкой.

Установить глобальную переменную окружения ITER\_NUM для вашего пользователя.

**Приложение Lab-02b:**

Создать программу идентичную программе Lab-02a, но:

* В первом вызове **CreateProcess** оставить только имя файла.
* В третьем вызове передать NULL вместо второго параметра, и перед этим установить локальную переменную окружения ITER\_NUM. Значение должно отличаться от глобальной переменной.

***Вопрос:*** *Все ли процессы созданы успешно? Сколько итераций выполнил каждый процесс и почему?*

**Приложение Lab-02hw:**

Создать программу, которая выводит "Hello, World!".

**Приложение Lab-02с:**

Создать программу, которая запускает 5 процессов для 5 копий приложения **Lab-02hw**. При запуске всех процессов используется только второй параметр функции **CreateProcess** и указывается только имя приложения без каких-либо путей к файлам.

В начале программы установить текущий каталог для программы **Lab-02с** в любую выбранную вами папку (в этой папке должно не должно быть файла **Lab-02с.exe**).

Перед тестированием программы создать и разместить 5 копий исполняемого файла **Lab-02hw.exe** следующим образом:

* **Lab-02hw-1.exe** в каталог, где находится файл **Lab-02с.exe** который будет запускаться при проверке работы программы;
* **Lab-02hw-2.exe** в каталог, который вы выбрали в качестве текущего для **Lab-02с**;
* **Lab-02hw-3.exe** в системный каталог Windows (можно заранее определить его с помощью функции [**GetSystemDirectory**](https://learn.microsoft.com/en-us/windows/desktop/api/sysinfoapi/nf-sysinfoapi-getsystemdirectorya))
* **Lab-02hw-4.exe** в основной каталог Windows (можно заранее определить его с помощью функции [**GetWindowsDirectory**](https://learn.microsoft.com/en-us/windows/desktop/api/sysinfoapi/nf-sysinfoapi-getwindowsdirectorya));
* **Lab-02hw-5.exe** в каталог, который находится в списке переменной **PATH** (можно использовать как существующий, так и добавить новый);

Проверить успешность запуска всех процессов.

**Приложения Lab-02inf и Lab-02d:**

Написать Lab-02inf с бесконечным циклом.

Написать Lab-02d, которое запускает Lab-02inf и ждёт ввода любого символа для продолжения работы. Изучить дескрипторы процесса Lab-02d в Process Explorer, найти дескриптор дочернего процесса.

После этого в аварийном режиме завершить дочерний процесс с помощью функции **TerminateProcess** и снова перевести программу в режим ожидания ввода любого символа. Изучить, что изменилось.

После этого попытаться получить и вывести PID дочернего процесса и вывести его на консоль. Программа снова ждёт ввода любого символа для продолжения работы.

***Вопрос:*** *Каковы результаты вывода на консоль и почему?*

Вызвать функцию **CloseHandle** для дочернего дескриптора и снова попытаться получить и вывести PID дочернего процесса. Изучить изменения в дескрипторах в программе Process Explorer. Программа снова ждёт ввода любого символа, после которого завершит свою работу.

**Постановка задачи для Linux:**

**Приложение Lab-02x:**

Программа является полным аналогом программы из заданий для Windows (модифицированной версии, не стартовой)

**Приложение Lab-02e:**

Программа создаёт два дочерних процесса и загружает в них исполняемый файл **Lab-02x**. При создании процессов должны быть выполнены следующие условия:

* В ходе запуска первого дочернего процесса параметр с количеством итераций должен быть передан как аргумент командной строки
* В ходе запуска второго дочернего процесса параметр с количеством итераций должен быть передан через переменную окружения ITER\_NUM. Переменная создается непосредственно в родительском процессе перед созданием второго дочернего процесса

Родительский процесс должен дождаться завершения всех дочерних процессов (выполняются одновременно).

Изучить информацию о созданных процессах через файловую систему (каталог /proc).

***Вопрос****: Все ли процессы созданы успешно? Если нет, почему?*

**Требования к выполненной работе:**

* Разработанные приложения должны быть скомпилированы и протестированы как на Windows, так и на ОС из семейства Linux.
* Для генерации и сборки проекта использовать средство CMake.
* Для компиляции файлов при сборке проекта использовать компилятор Clang (на обеих ОС: на Windows лучше установить через Visual Studio Installer).

**Дополнительно (совершенно не обязательно, но огромный плюс):**

1. Все операции копирования, задания переменных окружения вынести в конфигурацию CMake.
2. Создать кроссплатформенное приложение Lab-02-cross для создания единственного дочернего процесса Lab-02x с применением CMake. (например, логика и библиотеки выбираются для ОС через #ifdef и прочие директивы препроцессора). Количество итераций, как и раньше, либо через входные параметры, либо через переменные окружения, если входные параметры не указаны.

**Вопросы для контроля:**

1. Что такое процесс?
2. Какие ресурсы доступны процессу?
3. Что такое дочерний и родительский процесс?
4. Что такое системный вызов?
5. Какие системные вызовы предназначены для создания процессов в Windows? Linux?
6. Какие функции из WinAPI и POSIX связаны с ранее названными системными вызовами?
7. Сколько новых процессов появится в результате выполнения программой следующей последовательности вызовов **fork**() (учитывая, что все они выполнятся успешно)?

**fork**();

**fork**();

**fork**();

1. Для чего нужен **exec**?