Лебедевский Александр 3.11.2020

**Задание**

Приостановка и возобновление потоков

**Теория вопроса**

Каждый созданный поток имеет счетчик приостановок, максимальное значение

которого равно MAXIMUM\_SUSPEND\_COUNT. Счетчик приостановок показывает, сколько раз исполнение потока было приостановлено. Поток может исполняться только при условии, что значение счетчика приостановок равно нулю. В противном случае поток не исполняется. Исполнение каждого потока может быть приостановлено вызовом функции suspendThread, а именно:

DWORD SuspendThread(HANDLE hThread);

Эта функция увеличивает значение счетчика приостановок на 1. Для возобновления исполнения потока используется функция ResumeThread, которая имеет следующий прототип:

DWORD ResumeThread (HANDLE hThread) ;

Эта функция уменьшает значение счетчика приостановок на 1 при условии, что это значение было больше нуля. Если полученное значение счетчика приостановок равно 0, то исполнение потока возобновляется, в противном случае поток остается в приостановленном состоянии.

Поток может задержать свое исполнение вызовом функции sleep, которая имеет следующий прототип:

VOID Sleep(DWORD dwMilliseconds);

Единственный параметр функции sleep определяет количество миллисекунд, на которые поток, вызвавший эту функцию, приостанавливает свое исполнение. Если значение этого параметра равно 0, то выполнение потока просто прерывается, а затем возобновляется при условии, что нет других потоков, ждущих выделения процессорного времени. Если же значение этого параметра равно INFINITE, то поток приостанавливает свое исполнение навсегда, что приводит к блокированию работы приложения.

Поток можно завершить четырьмя способами:

1. функция потока возвращает управление;
2. поток самоуничтожается вызовом функции ExitThread;
3. один из потоков данного или стороннего процесса вызывает функцию TerminateThread;
4. завершается процесс, содержащий данный поток, следовательно, функция этого потока тоже завершается, не дойдя до своего конца.

Из всех этих (вышеперечисленных) способов только первый гарантирует правильное завершение программы, ведь перед выходом из функции вызываются деструкторы объектов, которые были использованы в этой функции, а в случае уничтожения потока с помощью второго, третьего и четвертого способов они вызваны не будут, так как функция не дойдет до своего конца. Поток может завершить сам себя принудительно, вызвав ExitThread(). Также поток может завершать другие потоки,используя функцию TerminateThread().

В процессе исследований необходимо:

1. Создать два потока main и thread
2. Проверить, что происходит, когда вызывается TerminateThread для потока main и как ведет себя при этом поток Thread
3. Выяснить роль дескрипторов потока

