МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное  
 учреждение высшего образования   
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ

**Отчет**

По лабораторной работе №4

на тему: **«Управление процессами в ОС Windows»**

по курсу

**«Операционные системы и системное программное обеспечение»**

Вариант №2

Выполнил:

студент гр. КТбо2-8

Жалнин Д.И.

Проверил:

Ассистент каф. МОП ЭВМ

Альминене Т. А.

# 1 Цель работы

Изучение средств управления процессами в ОС Windows, практическое овладение способами управления и синхронизации процессов и нитей.

2 Задание

Необходимо разработать программу, реализующую следующий алгоритм, используя Windows API:

Процесс порождает 10 дополнительных нитей, каждая из которых выдает на экран сообщение о своем запуске и свой номер. Далее в цикле основная нить задает пользователю вопрос, какую нить надо завершить. Когда завершены все дополнительные нити, завершается и весь процесс.

3 Ход работы

Для написания программы был выбран язык программирования С++. По ходу выполнения лабораторной работы был составлен следующий код:

#include <iostream>

#include <Windows.h>

int counter = 10;

HANDLE thread[10];

DWORD threadId[10];

int threadParams[10];

DWORD WINAPI threadFunc(LPVOID lpvThreadParam) {

int c = (\*((int\*)lpvThreadParam));

while (true) {

printf("%s %d %s", "Thread #", c, "is running\n");

Sleep(1000);

}

return(0);

}

void checkAlive() {

for (int i = 0; i < 10; i++)

if (thread[i]!=NULL && WAIT\_TIMEOUT != WaitForSingleObject(thread[i], 1)) {

thread[i] = NULL;

counter--;

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

for (int i = 0; i < counter; i++) {

threadParams[i] = i+1;

thread[i] = CreateThread(NULL, 0, &threadFunc, &threadParams[i], 0, &threadId[i]);

}

int threadNumber;

while (true) {

Sleep(10);

std::cout << "Введите номер останавливаемого потока(1-10) - ";

std::cin >> threadNumber;

if (threadNumber > 10 || threadNumber < 1 || thread[threadNumber-1] == NULL) {

std::cout << "Введены неверные данные или поток уже был остановлен\n";

} else {

TerminateThread(thread[threadNumber - 1], 0);

std::cout << "Поток #" << threadNumber << " был остановлен\n";

}

checkAlive();

std::cout << "На данный момент " << counter << " потоков работает\n";

if (counter <= 0) {

break;

}

}

}

Исходный код был написан и протестирован в среде разработки Visual Studio 2019. Программа решает поставленную задачу и отрабатывает без ошибок.

4 Пояснения

Программа состоит из трёх функций. Главная логика работы происходит в функции «main»: создание потоков, взаимодействие с ними и циклический вывод в консоль пользовательского интерфейса.

Создание потоков происходит в цикле, с занесением «хэндла» каждого потока в массив «thread» с помощью команды:

thread[i] = CreateThread(NULL, 0, &threadFunc, &threadParams[i], 0, &threadId[i]);

В функцию «CreateThread», которая возвращает «хэндл», передается функция «threadFunc», которая будет выполняться при запуске потока, и параметр, необходимый для вывода «приветствия» каждого потока в виде «Thread #X is running», где Х – целое число от 1 до 10.

Функция «threadFunc» принимает на вход указатель, в нашем случае на значение типа «int». Для уведомления пользователя о работе, функция каждую секунду выводит в консоль соответствующее уведомление.

После создания «пула» потоков, в цикле происходит взаимодействие с пользователем. Интерфейс пользователя сводится к завершению потока с индексом X, где X вводится пользователем в консоль. Поток с индексом Х завершается с помощью команды: «TerminateThread(thread[threadNumber - 1], 0);». На вход функции «TerminateThread» подается «Хэндл» потока, работу которого необходимо прервать и код его завершения.

После успешной или безуспешной попытки остановить поток, происходит проверка на «активность» всего пула потоков с помощью функции «checkAlive». Если при проходе пула функция обнаружила остановленный поток, она заменяет указатель на «хэндл» в пуле на NULL и уменьшает счетчик потоков на 1 (изначально счетчик установлен на 10). При обнулении счетчика программа завершает свою работу.

5 Выводы

В ходе выполнения лабораторной работы я научился работать со средствами управления потоками и процессами в операционной системе Windows с помощью «Windows API», которое можно использовать в большинстве языков программирования как подключаемую библиотеку. Полученные навыки помогают понять схему работы большинства многопоточных приложений, реализованных на язык программирования С, С++ и других для ОС Windows.