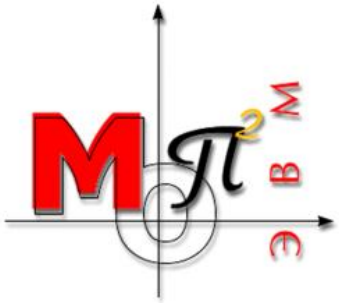
МИНОБРНАУКИ РОССИИ
Федеральное государственное автономное образовательное учреждения высшего образования
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт компьютерных технологий и информационной безопасности
Кафедра математического обеспечения и применения ЭВМ



ЛАБОРАТОРНАЯ РАБОТА № 4

по дисциплине
«Операционные системы и системное программное обеспечение»

на тему:
«Управление процессами в ОС Windows»
Вариант № 2

Выполнил:
Студент группы
КТбо2-8
Нестеренко П. А.

Проверила:
ассистент кафедры
МОП ЭВМ
Альминене Т. А.

Оценка

_____

«____» _____ 2020 г.

Таганрог 2020

1 ЦЕЛЬ РАБОТЫ

1.1 Дидактическая цель работы

Целью лабораторной работы является изучение средств управления процессами в ОС Windows, практическое овладение способами управления и синхронизации процессов и нитей.

1.2 Практическая цель работы

Необходимо разработать программу, реализующую поставленную задачу, используя Windows API.

2 ВАРИАНТ ЗАДАНИЯ

Задание. 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.

Решение. Для выполнения поставленной задачи использовался язык Си. В процессе выполнения лабораторной была написана следующая программа:

```
#define _CRT_SECURE_NO_WARNINGS
#include <windows.h>
#include <stdlib.h>
#include <locale.h>
#include <stdio.h>

#define NUMBER_OF_THREADS 10 // Количество создаваемыъ процессов
HANDLE hThreads[NUMBER_OF_THREADS];

DWORD WINAPI StartThread(CONST LPVOID lpParam)
{
    int c = (*((int*)lpParam)) + 1;
    printf("Нить под номером %d - запущена\n", c);
    while (true)
    {
        Sleep(100);
    }
    ExitThread(0);
}

void GetThreads()
{
    for (int i = 0; i < NUMBER_OF_THREADS; i++)
    {
        if (hThreads[i] != NULL && WaitForSingleObject(hThreads[i], 1))
            hThreads[i] = NULL;
        else printf("%d, ", i);
    }
}

int main(void)
{
    setlocale(LC_ALL, "Russian");
    DWORD                              dwThreadNumber[NUMBER_OF_THREADS],
dwThreadIds[NUMBER_OF_THREADS];

    for (int i = 0; i < NUMBER_OF_THREADS; i++)
    {
        Sleep(10);
        dwThreadNumber[i] = i;
        hThreads[i]    =    CreateThread(NULL,    0,    &StartThread,
&dwThreadNumber[i], 0, &dwThreadIds[i]);
        if (hThreads[i] == NULL)
            printf("Ошибка при создании потока: %d.\n", GetLastError());
    }

    printf("==================================\n");
    int Counter = NUMBER_OF_THREADS;
    while (Counter > 0)
    {
```

```
            Sleep(10);
            printf("Удалить поток под номером:");
            int NumberOfThread;
            scanf("%d", &NumberOfThread);

            if (NumberOfThread > 1 && NumberOfThread < NUMBER_OF_THREADS &&
hThreads[NumberOfThread - 1] != NULL)
            {
                TerminateThread(hThreads[NumberOfThread - 1], 0);
                printf("Был удалён поток с идентификатором: %d \n",
dwThreadIds[NumberOfThread - 1]);

                printf("Работающие потоки:");
                GetThreads();
                Counter--;
                printf("\n");
            }
            else printf("Введённые данные не корректны!!!\n");
        }
        printf("Все потоки удалены!\n");
        return 0;
    }
```

4 ПОЯСНЕНИЯ

Программа состоит из трёх функций, а именно:

1. **DWORD WINAPI StartThread(CONST LPVOID lpParam)** – функция, отвечающая за создание новых потоков.
2. **void GetThreads()** – функции проверяющая какие потоки ещё активны и выводящая их список на экран.
3. **int main(void)** – основная функция программы.

Функция main содержит цикл со счётчиком, в котором, с помощью команды hThreads[i] = CreateThread(NULL, 0, &StartThread, &dwThreadNumber[i], 0, &dwThreadIds[i]) создаётся 10 потоков(тредов). Если по какой-то причине создать поток не удалось, то будет выведено сообщение об ошибке. При создании нового потока вызывается функция StartThread. Она выводит порядковый номер созданного процесса и начинает бесконечный цикл while, в котором постоянно выполняется команда Sleep(100).

Далее в цикле while вводится номер потока для удаления. Удаление происходит с помощью функции TerminateThread (hThreads [NumberOfThread - 1] , 0), аргументами которой служат хендл потока и код возврата. После удаления потока вызывается функция GetThreads. Она выполняет обновления списка “активных” потоков, а так же выводит этот список, после чего счётчик цикла уменьшается. Работа программы продолжается до того момента, пока не будут удалены все созданные потоки.

3 ВЫВОД

В ходе выполнения лабораторной работы я познакомился со средствами управления потоками и процессами в операционной системе Windows с помощью «Windows API».Полученные навыки помогают понять схему работы большинства многопоточных приложений, реализованных на язык программирования C, C++ и других для ОС Windows.