МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 4

з дисципліни

«Операційні системи»

по темі :

“ Синхронізація потоків в ОС Windows ”

Виконав:

студент групи КН-209

Ярчак А.В

Викладач:

Кривенчук Ю. П.

Львів – 2019р.

Мета. Ознайомитися зі способами синхронізації потоків. Навчитися організовувати багатопоточність з використанням синхронізації за допомогою функцій WinAPI.

Завдання.

1. Реалізувати алгоритм із лабораторної роботи №3.
2. Здійснити розпаралелювання даного алгоритму на 2, 4, 8 потоків із використанням синхронізації.
3. Реалізувати прогрес (хід) виконання задачі.
4. Для синхронізації потоків використати такі методи: Interlocked-функції, м'ютекси.
5. Результати виконання роботи відобразити у звіті.

Код програми:

```
#include <iostream>
#include <windows.h>
#include <chrono>
#include <vector>

using namespace std;
using namespace std::chrono;
typedef struct SORT_PARAMS {
    int *ptr, size;
    double *time;
    SORT_PARAMS(int arr[], int s, double *t) : ptr(arr), size(s), time(t) {}
} THREAD_PARAMS;

HANDLE mutex = CreateMutexA(nullptr, FALSE, nullptr);

DWORD WINAPI Thread(LPVOID lpParam) {
    auto start = high_resolution_clock::now();
    int *arr = ((THREAD_PARAMS *) lpParam)->ptr;
    int size = ((THREAD_PARAMS *) lpParam)->size;
    int i, key, j;
    for (i = 0; i < size; i++) {
        WaitForSingleObject(mutex, INFINITE);
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] < key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
        Sleep(1);
        ReleaseMutex(mutex);
    }
    auto stop = high_resolution_clock::now();
    double *time = ((THREAD_PARAMS *) lpParam)->time;
    double buftime;
    *time = double(duration_cast<microseconds>(stop - start).count()) / 1000000;
    ::InterlockedExchangePointer(reinterpret_cast<void *volatile *>(&buftime),
                                 reinterpret_cast<void
*>(duration_cast<microseconds>(stop - start).count() / 1000000));
}

int main()
{
    cout << "=== An Array Top-Down Insert Sort ===" << endl;
    int size;
```

```
    cout << "Enter array size:";
    cin >> size;
    int arr[size];

    for (int i = 0; i < size; i++) arr[i] = rand() % 9000 + 1000;

    int count;
    cout << "Enter count of threads: ";
    cin >> count;

    cout << "Sort by " << count << " threads. Please wait..." << endl;

    HANDLE threads[count];
    double times[count];

    for (int i = 0; i < count; i++) {
        threads[i] = CreateThread(nullptr, 0, Thread, (PVOID) new
SORT_PARAMS(arr, size, &times[i]), 0, nullptr);
    }

    WaitForMultipleObjects(DWORD(count), threads, true, INFINITE);

    for (HANDLE i: threads) CloseHandle(i);
    for (int i = 0; i < count; i++)
        for (int k = i; k < count; k++)
            if (times[i] > times[k]) {
                double tp = times[i];
                times[i] = times[k];
                times[k] = tp;
            }

    for (double i: times) cout << i << ' ' << endl;
    for (int i : arr) cout << i << ' ';
    return 0;
}
```

Результат:

```
F:\Programs\CLion\Projects\OS\lab4\cmake-build-debug\lab4.exe
=== An Array Top-Down Insert Sort ===
Enter array size:100
100
Enter count of threads:4
 4
Sort by 4 threads. Please wait...
0.763074
0.765068
0.767027
0.768001
9962 9942 9723 9673 9500 9421 9308 9299 9035 8944 8827 8711 8667 8547 8541 8376 8118 7890 7868 7729 7724 7626 7464 7393
7370 7350 7334 7141 7084 7006 6811 6805 6771 6757 6705 6662 6604 6537 6447 6439 6436 6391 6281 5966 5931 5929 5827 5664
5648 5322 5190 5101 4902 4859 4726 4623 4548 4538 4382 4333 4316 4106 4035 3995 3942 3840 3538 3478 3358 3323 3082 3037
2954 2912 2895 2869 2842 2718 2703 2629 2264 2253 2169 2145 1961 1894 1778 1756 1741 1716 1644 1529 1491 1467 1446 1292
1288 1153 1041 1040
Process finished with exit code 0
```

Висновок: на цій лабораторній роботі я навчитися організовувати багатопоточність з використанням синхронізації за допомогою функцій WinAPI. Я синхронізував паралельне сортування за допомогою м’ютексів та Interlocked-функції. На виході отримав повністю відсортований масив цілих чисел.