

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет "Львівська політехніка"
Кафедра САПР



Звіт з лабораторної роботи №7
з дисципліни "Об'єктно-орієнтоване
програмування"
"API-функції для управління потоками та процесами.
Синхронізація потоків у Windows."
Варіант - 11

Виконав:
ст.гр. КН-108
Пагута В.О.
Прийняв:
Головатий А.І.

Львів 2022

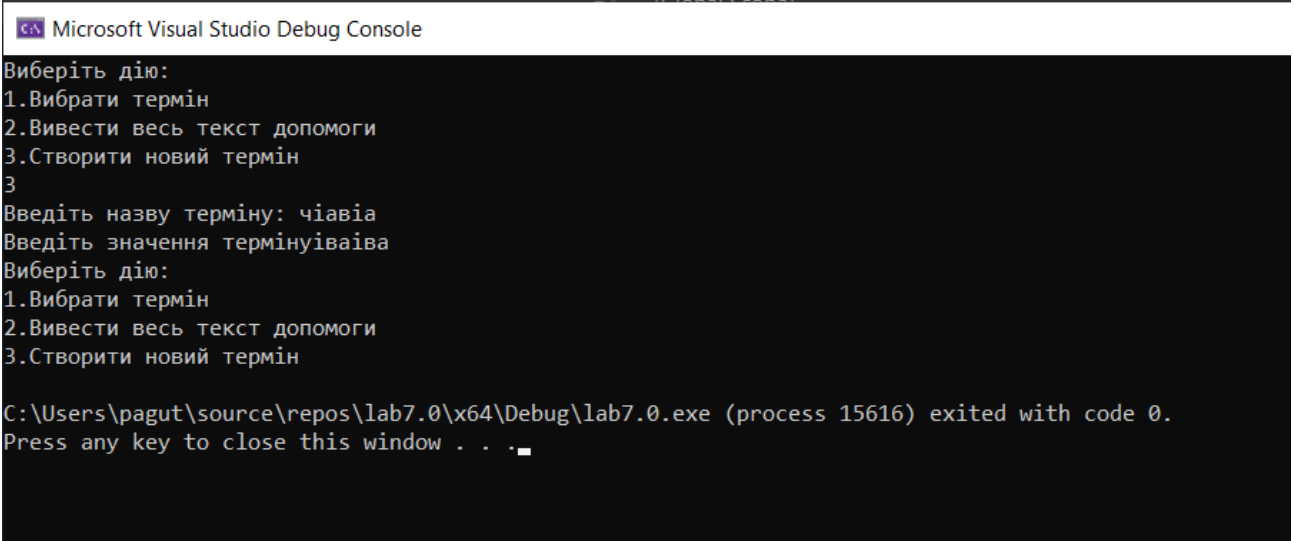
Мета роботи:

Ознайомитись з основними API-функціями для управління потоками та процесами, а також з методикою синхронізації потоків за допомогою критичних секцій.

Завдання №1.1

```
#include <iostream>
#include <Windows.h>
#include <string>
#include <tchar.h>
using namespace std;
class Process {
public:
    Process() {
        STARTUPINFO sti;
        ZeroMemory(&sti, sizeof(STARTUPINFO));
        sti.cb = sizeof(STARTUPINFO);
        PROCESS_INFORMATION pi;
        LPTSTR szCmdline =
            _tcsdup(TEXT("C:\\Users\\pagut\\source\\repos\\Project15\\x64\\Debug\\Project15.exe"));
        ZeroMemory(&pi, sizeof(pi));
        if (CreateProcess(NULL, szCmdline, NULL, NULL, TRUE,
            NORMAL_PRIORITY_CLASS, NULL, NULL, &sti, &pi)) {
            Sleep(11000);
            TerminateProcess(pi.hProcess, 0);
            CloseHandle(pi.hProcess);
            CloseHandle(pi.hThread);
        }
        else
            cout << "Couldn't start process" << endl;
    }
};
int main() {
    Process docxFile;
}
```

Виконання:



```
Microsoft Visual Studio Debug Console
Виберіть дію:
1.Вибрати термін
2.Вивести весь текст допомоги
3.Створити новий термін
3
Введіть назву терміну: чіавіа
Введіть значення термінуіваіва
Виберіть дію:
1.Вибрати термін
2.Вивести весь текст допомоги
3.Створити новий термін
C:\Users\pagut\source\repos\lab7.0\x64\Debug\lab7.0.exe (process 15616) exited with code 0.
Press any key to close this window . . .
```

Завдання №1.2:

11. Написати програму для одночасного опрацювання масиву двома потоками. Перший потік знаходитиме кількість елементів масиву, які більші за 1234, другий - середнє арифметичне непарних додатних елементів масиву.

```
#include <iostream>
#include <Windows.h>
using namespace std;
int BigNumbers(int* arr, int size)
{
    int result = 0;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] > 1234)
        {
            result++;
        }
    }
    return result;
}
float Arithmetic(int* arr, int size)
{
    float result = 0;
    int amount = 0;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] > 0 && arr[i] % 2 != 0)
        {
            result += arr[i];
            amount++;
        }
    }
    result = result / amount;
    return result;
}
class Data {
public:
    int* arr;
    int size;
    Data(int* arr, int size) {
        this->arr = arr;
        this->size = size;
    }
};
DWORD WINAPI thread1(LPVOID lpParameter) {
    Data* instance = (Data*)lpParameter;
    int counter = BigNumbers(instance->arr, instance->size);
    cout << "Кількість елементів масиву, більші за 1234 = " << counter << endl;
    return 0;
}
DWORD WINAPI thread2(LPVOID lpParameter) {
    Data* instance = (Data*)lpParameter;
    float avarage = Arithmetic(instance->arr, instance->size);
    cout << "Середнє арифметичне непарних додатних елементів масиву = " << avarage
<< endl;
    return 0;
}
class Process {
    HANDLE handle1;
    HANDLE handle2;
public:
    Process(int* arr, int size) {
        Data data(arr, size);
        this->handle1 = CreateThread(0, 0, thread1, &data, 0, 0);
        this->handle2 = CreateThread(0, 0, thread2, &data, 0, 0);
        WaitForSingleObject(handle1, INFINITE);
        WaitForSingleObject(handle2, INFINITE);
    }
};
```

```

        CloseHandle(handle1);
        CloseHandle(handle2);
    }
};
int main(void) {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    srand(time(0));
    int size;
    cout << "Введіть розмір масиву: ";
    cin >> size;
    int* arr = new int[size];
    for (int i = 0; i < size; i++)
    {
        arr[i] = rand() % 4000 - 2000;
        cout << arr[i] << " ";
    }
    cout << endl;
    Process instance(arr, size);
}

```

Виконання:

 Microsoft Visual Studio Debug Console

```

Введіть розмір масиву: 6
-1781 847 1593 1542 658 1640
Кількість елементів масиву, більші за 1234 = 3
Середнє арифметичне непарних додатніх елементів масиву = 1220

C:\Users\pagut\source\repos\lab7.1_API\x64\Debug\lab7.1_API.exe (process 27192) exited with code 0.
Press any key to close this window . . .

```

Завдання №2:

11. Написати програму для одночасного опрацювання матриці двома потоками. Перший потік замінюватиме кожен парний елемент масиву на значення другого елемента поточного рядка, другий - знаходитиме кількість рядків, у яких принаймні один елемент дорівнює другому елементу даного рядка.

```
#include <iostream>
#include <Windows.h>
#include <iomanip>
#define height 5
#define width 5
using namespace std;
DWORD WINAPI thread1(LPVOID lpParameter);
DWORD WINAPI thread2(LPVOID lpParameter);
void MatrixOutput(int** matrix)
{
    for (size_t i = 0; i < height; i++)
    {
        for (size_t j = 0; j < width; j++)
        {
            if (matrix[i][j] < 0)
            {
                cout << matrix[i][j] << "\t";
            }
            else
                cout << " " << matrix[i][j] << "\t";
        }
        cout << endl;
    }
}
void ChangeMatrix(int** matrix)
{
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            if (j == 1) {
                continue;
            }
            if (matrix[i][j] % 2 == 0) {
                matrix[i][j] = matrix[i][1];
            }
        }
    }
}
int CounterLines(int** matrix)
{
    int result = 0;
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            if (j == 1)
            {
                continue;
            }
            if (matrix[i][j] == matrix[i][1])
            {
                result++;
                break;
            }
        }
    }
}
```

```

        return result;
    }
class Process {
    HANDLE handle1;
    HANDLE handle2;
public:
    Process(int** matrix) {
        this->handle1 = CreateThread(0, 0, thread1, matrix, 0, 0);
        this->handle2 = CreateThread(0, 0, thread2, matrix, 0, 0);
        WaitForSingleObject(handle1, INFINITE);
        WaitForSingleObject(handle2, INFINITE);
        CloseHandle(handle1);
        CloseHandle(handle2);
    }
};
DWORD WINAPI thread1(LPVOID lpParameter) {
    int** arr = (int**)lpParameter;
    ChangeMatrix(arr);
    MatrixOutput(arr);
    return 0;
}
DWORD WINAPI thread2(LPVOID lpParameter) {
    int** arr = (int**)lpParameter;
    int counter = CounterLines(arr);
    cout << "Кількість рядків які мають щонайменше 1 елемент, який == 2 елементу
поточного рядка : " << counter << endl;
    return 0;
}
int main(void) {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    srand(time(0));
    int lines = 0;
    int** matrix = new int*[height];
    for (int i = 0; i < height; i++)
    {
        matrix[i] = new int[width];
        for (int j = 0; j < width; j++)
        {
            matrix[i][j] = rand() % 20 - rand() % 20;
        }
    }
    cout << "Звичайна матриця:" << endl;
    MatrixOutput(matrix);
    cout << endl;
    Process instance(matrix);
}
#include <iostream>
#include <Windows.h>
#include <iomanip>
#define height 5
#define width 5
using namespace std;
DWORD WINAPI thread1(LPVOID lpParameter);
DWORD WINAPI thread2(LPVOID lpParameter);
void MatrixOutput(int** matrix)
{
    for (size_t i = 0; i < height; i++)
    {
        for (size_t j = 0; j < width; j++)
        {
            if (matrix[i][j] < 0)
            {
                cout << matrix[i][j] << "\t";
            }
            else
                cout << " " << matrix[i][j] << "\t";
        }
        cout << endl;
    }
}

```

```

    }
}
void ChangeMatrix(int** matrix)
{
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            if (j == 1) {
                continue;
            }
            if (matrix[i][j] % 2 == 0) {
                matrix[i][j] = matrix[i][1];
            }
        }
    }
}
int CounterLines(int** matrix)
{
    int result = 0;
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            if (j == 1)
            {
                continue;
            }
            if (matrix[i][j] == matrix[i][1])
            {
                result++;
                break;
            }
        }
    }
    return result;
}
class Process {
    HANDLE handle1;
    HANDLE handle2;
public:
    Process(int** matrix) {
        this->handle1 = CreateThread(0, 0, thread1, matrix, 0, 0);
        this->handle2 = CreateThread(0, 0, thread2, matrix, 0, 0);
        WaitForSingleObject(handle1, INFINITE);
        WaitForSingleObject(handle2, INFINITE);
        CloseHandle(handle1);
        CloseHandle(handle2);
    }
};
DWORD WINAPI thread1(LPVOID lpParameter) {
    int** arr = (int**)lpParameter;
    ChangeMatrix(arr);
    MatrixOutput(arr);
    return 0;
}
DWORD WINAPI thread2(LPVOID lpParameter) {
    int** arr = (int**)lpParameter;
    int counter = CounterLines(arr);
    cout << "Кількість рядків які мають щонайменше 1 елемент, який == 2 елементу  

поточного рядка : " << counter << endl;
    return 0;
}
int main(void) {
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    srand(time(0));
}

```

```

int lines = 0;
int** matrix = new int*[height];
for (int i = 0; i < height; i++)
{
    matrix[i] = new int[width];
    for (int j = 0; j < width; j++)
    {
        matrix[i][j] = rand() % 20 - rand() % 20;
    }
}
cout << "Звичайна матриця:" << endl;
MatrixOutput(matrix);
cout << endl;
Process instance(matrix);
}

```

Виконання:

Microsoft Visual Studio Debug Console

Звичайна матриця:

```

10      8      -6      -17     -10
6        7      -1        5        9
1       -8        2      -10        1
-4        7        6      -11       11
-2      -14     -13        6       -9

8        8        8      -17        8
7        7      -1        5        9
1       -8      -8       -8        1
7        7        7      -11       11
-14     -14     -13     -14       -9

```

Кількість рядків які мають щонайменше 1 елемент, який == 2 елементу поточного рядка : 5

C:\Users\pagut\source\repos\lab7.2_API\x64\Debug\lab7.2_API.exe (process 29196) exited with code 0.
Press any key to close this window . . .

Контрольні запитання:

1. Що таке процес?

Процес — об'єкт операційної системи, контейнер системних ресурсів, призначених для підтримки виконання програми.

2. Яка API-функція використовується для створення процесу?

Базовою функцією API для створення процесу є `CreateProcess`.

3. Яка API-функція використовується для завершення процесу?

Процес виконує останню інструкцію програми — повертає операційній системі код завершення. Якщо процес завершився нормально повертається значення 0, інакше повертається значення коду помилки.

операційна система встановлює стан процесу «завершений» і починає звільнення ресурсів, які були виділені процесу під час його виконання

операційна система по черзі завершує усі дочірні процеси

даного операційна система звільняє адресний простір

процесу

операційна система усуває процес з черги готових процесів.

4. Що таке потік?

спосіб програми розділити себе на дві чи більше паралельні задачі.

5. Що таке синхронізація потоків?

Виконання функцій, зазначених потокам одночасно.

6. Які є види синхронізації?

Найпоширеніші засоби синхронізації такі:

- [сигнали](#) і [повідомлення](#),
- [критичні секції](#),
- [м'ютекси](#)
- [семафори](#)
- [події](#)
- [бар'єри](#),
- канали ([англ. pipe](#)).

8. Що таке об'єкт синхронізації «подія»?

Подія (об'єкт події, event object) в операційній системі Windows — об'єкт для синхронізації виконання процесів (потоків), який може знаходитися у двох станах (сигнальному та несигнальному).

Висновок:

Я ознайомився з основними API-функціями для управління потоками та процесами, а також з методикою синхронізації потоків за допомогою критичних секцій.