МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 2

з дисципліни

«Операційні системи»

по темі :

“ Робота з процесами в ОС Windows ”

Виконав:

студент групи КН-209

Ярчак А.В

Викладач:

Кривенчук Ю. П.

Львів – 2019р.

Мета. Ознайомитися з багатопоточністю в ОС Windows. Навчитися працювати з процесами, використовуючи WinAPI-функції.

Завдання.

1. Створити окремий процес, і здійснити в ньому табулювання функції, задану розкладом в ряд Тейлора, в області її визначення на відрізку від A до B (кількість кроків не менше 100 000). Функцію взяти з у відповідності до номера функції та порядкового номера у журнальному списку
2. Реалізувати табулювання функцій у 2-ох, 4-ох, 8-ох процесах. Для кожного процесу реалізувати можливість його запуску, зупинення, завершення та примусове завершення («вбиття»). Виміряти час роботи процесів за допомогою функцій WinAPI. Порівняти результати роботи в одному і в багатьох процесах.
3. Реалізувати можливість зміни пріоритету виконання процесу.
4. Продемонструвати результати виконання роботи, а також кількість створених процесів у “Диспетчері задач”, або подібних утилітах (н-д, ProcessExplorer)/

Індивідуальні завдання.

$$arctg(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \ldots + (-1)^n \frac{x^{2n+1}}{2n+1} + \ldots, |x| < 1$$

Код програми з табулюванням:

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
	double x = 1, arctg = 0, y = 0;
	int A = 19, B = 171215;
	for (int n = A; n < B; n++)
	{
		y = pow(-1, n) * (pow(x, 2 * n + 1) / (2 * n + 1));
		cout << y << endl;
		arctg += y;
	}
	cout << arctg << endl;
	return 0;
}
```

Код програми:

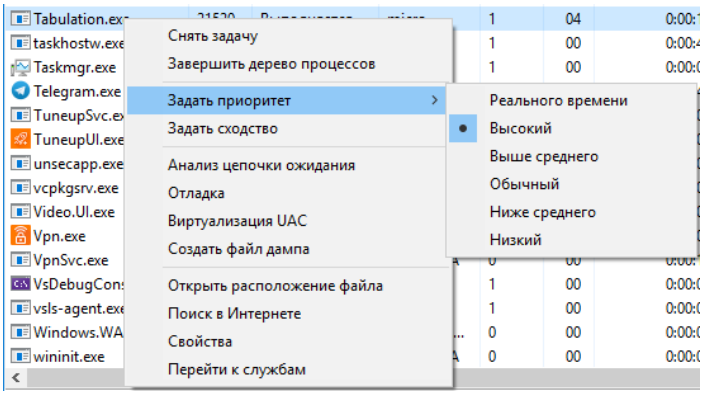
```
#include <iostream>
#include <tchar.h>
#include <ctime>
#include <Windows.h>
using namespace std;

PROCESS_INFORMATION processes[32];
DWORD priority[6] = { REALTIME_PRIORITY_CLASS, HIGH_PRIORITY_CLASS, ABOVE_NORMAL_PRIORITY_CLASS,
NORMAL_PRIORITY_CLASS, BELOW_NORMAL_PRIORITY_CLASS, IDLE_PRIORITY_CLASS };

int main()
{
	int count;
	cout << "Enter a count of processes: ";
	cin >> count;
	for (int i = 0; i < count; i++)
	{
		STARTUPINFO si;
		PROCESS_INFORMATION pi;
		ZeroMemory(&si, sizeof(STARTUPINFO));
		si.cb = sizeof(STARTUPINFO);
```

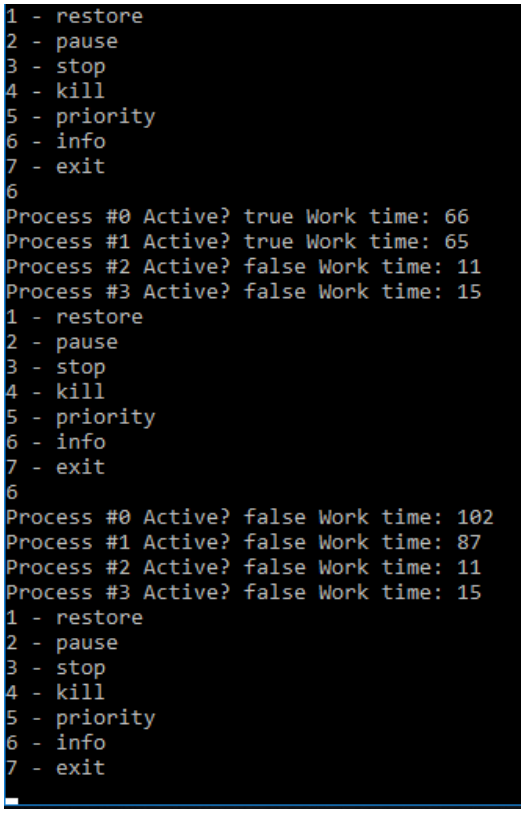
```
		ZeroMemory(&pi, sizeof(STARTUPINFO));
		TCHAR name[] = "C:/Users/micro/source/repos/Tabulation/Debug/Tabulation.exe";

		if (CreateProcess(name, NULL, NULL, NULL, 0, CREATE_NEW_CONSOLE, NULL, NULL, &si, &pi))
		{
			cout << "Process " << i << " was created!" << endl;
			processes[i] = pi;
		}
		else
			cout << "Error! " << GetLastError() << endl;
	}
	while (true)
	{
		int num, action;
		DWORD flag, exit_code = 0;;
		cout << "1 - restore" << endl << "2 - pause" << endl << "3 - stop" << endl << "4 - kill" <<
endl << "5 - priority"<< endl << "6 - info" << endl << "7 - exit" << endl;
		cin >> action;
		switch (action)
		{
			case 1:
				cout << "Select the process number: ";
				cin >> num;
				flag = ResumeThread(processes[num].hThread);
				if (flag > DWORD(1))
					cout << "The selected process are suspended!" << endl;
				else if (flag == DWORD(1))
					cout << "The selected process was restored!" << endl;
				else if (flag == DWORD(0))
					cout << "The selected process was not stopped!" << endl;
				else if (flag == DWORD(-1))
					cout << "Error! "<< GetLastError() << endl;
				break;
			case 2:
				cout << "Select the process number: ";
				cin >> num;
				if (SuspendThread(processes[num].hThread) != DWORD(-1))
					cout << "The selected process was suspended! " << num << endl;
				else
					cout << "Error! " << GetLastError() << endl;
				break;
			case 3:
				cout << "Select the process number: ";
				cin >> num;
				TerminateThread(processes[num].hThread, 0);
				break;
			case 4:
				cout << "Select the process number: ";
				cin >> num;
				if (TerminateProcess(processes[num].hProcess, 0) != 0)
					cout << "Successful!" << endl;
				else
					cout << "Error! " << GetLastError() << endl;
				break;
			case 5:
				cout << "Select the process number: ";
				cin >> num;
				int choice;
				cout << "  1 - Real Time priority" << endl << "  2 - High priority" << endl
<< "  3 - Above Normal priority" << endl << "  4 - Normal priority" << endl << "  5 - Below Normal priority"
<< endl << "  6 - Idle priority" << endl;
				cin >> choice;
				if(SetPriorityClass(processes[num].hProcess, priority[choice - 1]) != 0)
					cout << "Successful!" << endl;
				else
					cout << "Error! " << GetLastError() << endl;
				break;
			case 6:
				for (int i = 0; i < count; i++)
				{
					FILETIME ft[4];
					DWORD isActive = 0;
					double vars[2];
					GetExitCodeProcess(processes[i].hProcess, &isActive);
					if (GetProcessTimes(processes[i].hProcess, &ft[0], &ft[1], &ft[2],
&ft[3]) != -1)
					{
						cout << "Process #" << i << " Active? " << ((isActive ==
STILL_ACTIVE)?"true":"false");
						for (int k = 0; k < 2; k++)
```

```
                                                {
                                                        LARGE_INTEGER date;
                                                        date.HighPart = ft[k].dwHighDateTime;
                                                        date.LowPart = ft[k].dwLowDateTime;
                                                        vars[k] = (date.QuadPart - 11644473600000 * 10000) /
10000000;
                                                }
                                                cout << " Work time: " << ((isActive == STILL_ACTIVE) ?
(time(nullptr) - vars[0]) : (vars[1] - vars[0])) << endl;
                                        }
                                }
                                break;
                        case 7:
                                goto link;
                                break;
                        case 666:
                                for (int i = 0; i < count; i++)
                                        TerminateProcess(processes[i].hProcess, 0);
                                break;
                }
        }
        link:
        return 0;
}
```

Результат:

```
C:\Users\micro\source\repos\OS\lab1\OS\Debug\OS.exe
Enter a count of processes: 4
Process 0 was created!
Process 1 was created!
Process 2 was created!
Process 3 was created!
1 - restore
2 - pause
3 - stop
4 - kill
5 - priority
6 - info
7 - exit
2
Select the process number: 0
The selected process was suspended! 0
1 - restore
2 - pause
3 - stop
4 - kill
5 - priority
6 - info
7 - exit
3
Select the process number: 2
1 - restore
2 - pause
3 - stop
4 - kill
5 - priority
6 - info
7 - exit
4
Select the process number: 3
Successful!
1 - restore
2 - pause
3 - stop
4 - kill
5 - priority
6 - info
7 - exit
1
Select the process number: 0
The selected process was restored!
1 - restore
2 - pause
3 - stop
4 - kill
5 - priority
6 - info
7 - exit
5
Select the process number: 1
  1 - Real Time priority
  2 - High priority
  3 - Above Normal priority
  4 - Normal priority
  5 - Below Normal priority
  6 - Idle priority
2
Successful!
```





Висновок: на цій лабораторній роботі я навчився працювати з процесами за допомогою WinAPI функцій: зупиняти, відновлювати, зупиняти, вбивати процес. Також змінювати приорітет процесу та вимірювати час його роботи.