Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра кібербезпеки та програмного забезпечення

**Звіт**

з виконаної лабораторної роботи № 1

Дисципліна: Системне програмування

на тему

«Процеси, потоки, базова структура ПЗ»

Виконав :

студент академічної групи КІ-15

Аннаєв А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив :

Викладач

КонстантиноваЛ.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

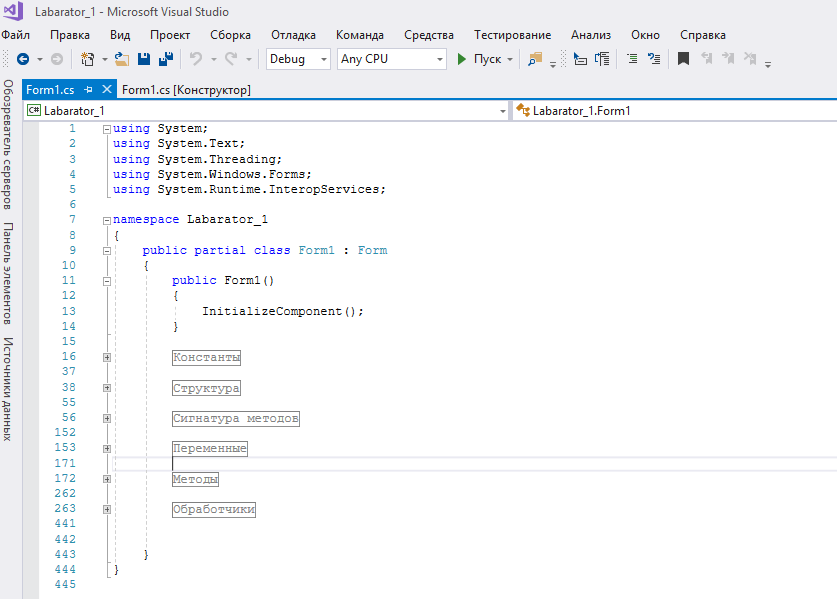
Кропивницкий- 2018

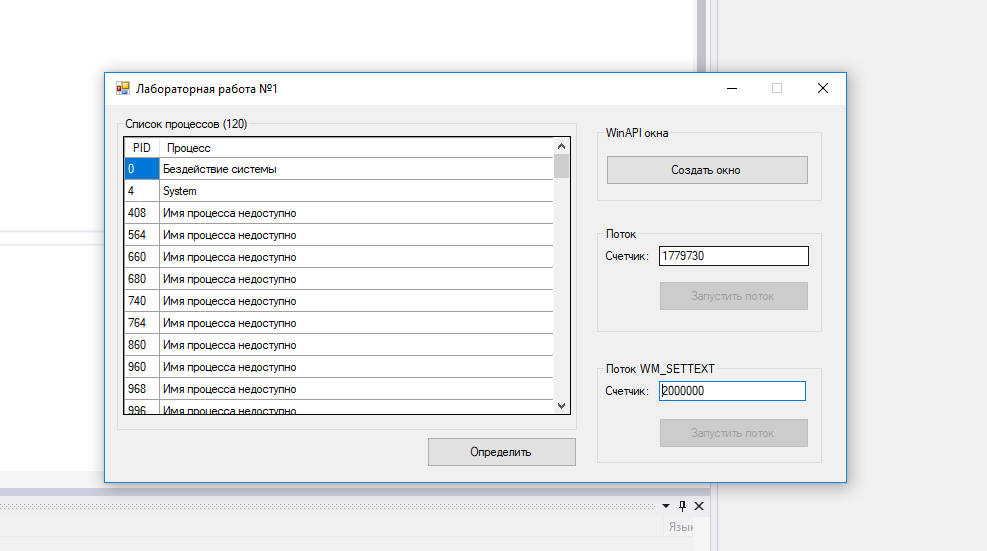
**Мета:** Отримати практичні навики використання функцій Win API.

**Завдання:**

Використовуючи наявну електронну документацію повністю повторити функціонал програм наведених в прикладах на мові С++ чи С#.

Тобто провести перетворення програм (трансляцію), представлених на мові програмування Object Pascal на мову С++ чи С#.





using System;

using System.Text;

using System.Threading;

using System.Windows.Forms;

using System.Runtime.InteropServices;

namespace Labarator\_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

#region Константы

const int PROCESS\_QUERY\_INFORMATION = 0x0400;

const int PROCESS\_VM\_READ = 0x0010;

const int WS\_CAPTION = 0xC00000;

const int WS\_CHILD = 0x40000000;

const int WS\_MINIMIZEBOX = 0x20000;

const int WS\_TABSTOP = 0x10000;

const int WS\_SYSMENU = 0x80000;

const int WS\_VISIBLE = 0x10000000;

const int WS\_EX\_CLIENTEDGE = 0x00000200;

const int BS\_DEFPUSHBUTTON = 0x0010;

const int WM\_COMMAND = 0x0111;

const int WM\_CLOSE = 0x0010;

const int WM\_SETTEXT = 0x000C;

const int SS\_CENTER = 0x0001;

const int CBS\_DROPDOWNLIST = 0x0003;

const int CB\_ADDSTRING = 0x0143;

const int CB\_SETCURSEL = 0x014E;

#endregion

#region Структура

struct WNDCLASS

{

public uint style;

public WndProcDelegate lpfnWndProc;

public int cbClsExtra;

public int cbWndExtra;

public IntPtr hInstance;

public IntPtr hIcon;

public IntPtr hCursor;

public IntPtr hbrBackground;

public string lpszMenuName;

public string lpszClassName;

}

#endregion

#region Сигнатура методов

/// <summary>

/// Получает полное имя исполняемого образа для указанного процесса.

/// </summary>

[DllImport("kernel32.dll")]

private static extern bool QueryFullProcessImageName(IntPtr hProcess, uint dwFlags, StringBuilder lpExeName, ref uint lpdwSize);

/// <summary>

/// Закрытие дескриптора

/// </summary>

/// <param name="handle">дескриптор</param>

/// <returns></returns>

[DllImport("kernel32.dll")]

private static extern bool CloseHandle(IntPtr handle);

/// <summary>

/// Опредение перечисления процессов

/// </summary>

[DllImport("psapi.dll")]

private static extern bool EnumProcesses(uint[] processIds, uint arraySizeBytes, out uint bytesCopied);

/// <summary>

/// Открывает существующий локальный объект процесса.

/// </summary>

[DllImport("kernel32.dll")]

private static extern IntPtr OpenProcess(int dwDesiredAccess, bool bInheritHandle, uint dwProcessId);

/// <summary>

/// Регистрирует класс окна для последующего использования при вызовах функции

/// </summary>

[DllImport("user32.dll")]

private static extern UInt16 RegisterClass(ref WNDCLASS lpWndClass);

/// <summary>

/// Отображает модальное диалоговое окно

/// </summary>

[DllImport("user32.dll")]

private static extern int MessageBox(IntPtr hWnd, String text, String caption, int options);

/// <summary>

/// Вызывает оконную процедуру по умолчанию для предоставления обработки по умолчанию для любых оконных сообщений

/// </summary>

[DllImport("user32.dll")]

private static extern IntPtr DefWindowProc(IntPtr hWnd, uint msg, IntPtr wParam, IntPtr lParam);

/// <summary>

/// Уничтожает указанное окно

/// </summary>

[DllImport("user32.dll")]

private static extern bool DestroyWindow(IntPtr hWnd);

/// <summary>

/// Отображение окна

/// </summary>

[DllImport("user32.dll")]

private static extern bool ShowWindow(IntPtr hWnd, int nCmdShow);

/// <summary>

/// Получает псевдо-дескриптор для текущего процесса.

/// </summary>

[DllImport("kernel32.dll")]

private static extern IntPtr GetCurrentProcess();

/// <summary>

/// Отпрака сообщение

/// </summary>

[DllImport("user32.dll")]

private static extern int SendMessage(IntPtr hwnd, int msg, IntPtr wParam, string lParam);

/// <summary>

/// Копирует текст строки заголовка указанного окна (если он есть) в буфер

/// </summary>

[DllImport("user32.dll")]

private static extern int GetWindowText(IntPtr hWnd, StringBuilder lpString, int nMaxCount);

/// <summary>

/// Изменяет текст строки заголовка указанного окна (если он есть).

/// </summary>

[DllImport("user32.dll")]

private static extern bool SetWindowText(IntPtr hwnd, String lpString);

/// <summary>

/// Создание окна

/// </summary>

[DllImport("user32.dll")]

private static extern IntPtr CreateWindowEx(UInt32 dwExStyle,

string lpClassName,

string lpWindowName,

UInt32 dwStyle,

Int32 x, Int32 y, Int32 nWidth, Int32 nHeight, IntPtr hWndParent,

IntPtr hMenu,

IntPtr hInstance,

IntPtr lpParam);

#endregion

#region Переменные

private IntPtr hTextBox;

private static IntPtr hShowDialog;

private static IntPtr hAbout;

private static IntPtr hExit;

private static IntPtr hText;

private static IntPtr hDialog;

private static IntPtr hdOk;

private static IntPtr hdCancel;

private static IntPtr hdHelp;

private static IntPtr hdCombo;

private static IntPtr hParam;

private static IntPtr hdText;

delegate IntPtr WndProcDelegate(IntPtr hWnd, uint msg, IntPtr wParam, IntPtr lParam);

delegate void SetTextCallback(string text);

#endregion

#region Методы

/// <summary>

/// Определение перечисления процессов

/// </summary>

private void GetEnumProcess()

{

uint arraySize = 120;

uint[] processIds = new uint[arraySize];

uint bytesCopied;

EnumProcesses(processIds, arraySize \* 4, out bytesCopied);

uint numIdsCopied = bytesCopied >> 2;

groupBox1.Text += " (" + numIdsCopied.ToString() + ")";

IntPtr hProcess;

dataGridView1.RowCount = (int)numIdsCopied;

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

{

hProcess = OpenProcess(PROCESS\_QUERY\_INFORMATION | PROCESS\_VM\_READ, false, processIds[i]);

StringBuilder processName = new StringBuilder(255);

uint hz = 255;

QueryFullProcessImageName(hProcess, 0, processName, ref hz);

dataGridView1[0, i].Value = processIds[i].ToString();

if (processName.Length == 0)

{

if (processIds[i] == 0)

processName.Append("Бездействие системы");

else if (processIds[i] == 4)

processName.Append("System");

else

processName.Append("Имя процесса недоступно");

}

dataGridView1[1, i].Value = processName.ToString();

CloseHandle(hProcess);

}

}

/// <summary>

/// Метод запускаемый в отдельном потоке

/// </summary>

private void FirstThread()

{

for (int i = 0; i <= 2000000; i++)

{

if (i % 10 == 0)

SetText(i.ToString());

}

}

/// <summary>

/// Метод запускаемый в отдельном потоке

/// </summary>

private void SecondThread()

{

//Вывод сообщения с потока на главную форму

for (int i = 0; i <= 2000000; i++)

{

if (i % 10 == 0)

SendMessage(hTextBox, WM\_SETTEXT, IntPtr.Zero, i.ToString());

}

}

/// <summary>

/// Установка текста

/// </summary>

/// <param name="text"></param>

private void SetText(string text)

{

if (textBox1.InvokeRequired)

{

SetTextCallback d = new SetTextCallback(SetText);

try

{

Invoke(d, new object[] { text });

}

catch { }

}

else

textBox1.Text = text;

}

#endregion

#region Обработчики

/// <summary>

/// Обработчик кнопки определения процессов

/// </summary>

private void btnDefine\_Click(object sender, EventArgs e)

{

GetEnumProcess();

}

/// <summary>

/// Обработчик кнопки запуска потока

/// </summary>

private void btnStartThread1\_Click(object sender, EventArgs e)

{

btnStartThread1.Enabled = false;

Thread firstThread = new Thread(new ThreadStart(FirstThread));

firstThread.Start();

}

/// <summary>

/// Обработчик кнопки запуска потока

/// </summary>

private void btnStartThread2\_Click(object sender, EventArgs e)

{

btnStartThread2.Enabled = false;

hTextBox = textBox2.Handle;

Thread secondThread = new Thread(new ThreadStart(SecondThread));

secondThread.Start();

}

/// <summary>

/// Создание окна WinApi

/// </summary>

private void btnCreateWindow\_Click(object sender, EventArgs e)

{

string className = "window";

WNDCLASS wndclass = new WNDCLASS();

wndclass.lpszClassName = className;

wndclass.lpfnWndProc = CustomWndProc;

RegisterClass(ref wndclass);

// главное окно

IntPtr hWindow = CreateWindowEx(0, className, "Окно",

WS\_CAPTION | WS\_MINIMIZEBOX | WS\_SYSMENU | WS\_VISIBLE,

Location.X + 100, Location.Y + 100,

300, 150, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hShowDialog = CreateWindowEx(0, "BUTTON", "Диалог",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

25, 80, 75, 23,

hWindow, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hAbout = CreateWindowEx(0, "BUTTON", "Про окно",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

105, 80, 75, 23, hWindow,

IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hExit = CreateWindowEx(0, "BUTTON", "Выход",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

185, 80, 75, 23,

hWindow, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hText = CreateWindowEx(0, "STATIC", "Откройте диалог,",

WS\_CHILD | WS\_VISIBLE | SS\_CENTER,

10, 10, 265, 20,

hWindow, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hParam = CreateWindowEx(0, "STATIC", "чтоб ввести текст и указать параметр",

WS\_CHILD | WS\_VISIBLE | SS\_CENTER,

10, 40, 265, 20, hWindow,

IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

}

/// <summary>

/// Обработчик сообщений окна

/// </summary>

private static IntPtr CustomWndProc(IntPtr hWnd, uint msg, IntPtr wParam, IntPtr lParam)

{

if (msg == WM\_COMMAND)

{

if (lParam == hShowDialog)

{ // диалог

hDialog = CreateWindowEx(0, "window", "Диалог",

WS\_CAPTION | WS\_MINIMIZEBOX | WS\_SYSMENU | WS\_VISIBLE,

500, 500, 300, 200,

hWnd, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hdOk = CreateWindowEx(0, "BUTTON", "Окай",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

195, 10, 75, 23,

hDialog, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hdCancel = CreateWindowEx(0, "BUTTON", "Отмена",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

195, 40, 75, 23,

hDialog, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hdHelp = CreateWindowEx(0, "BUTTON", "Справка",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

195, 70, 75, 23,

hDialog, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hdText = CreateWindowEx(WS\_EX\_CLIENTEDGE, "EDIT", "Текст",

WS\_CHILD | BS\_DEFPUSHBUTTON | WS\_TABSTOP | WS\_VISIBLE,

10, 10, 175, 23,

hDialog, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

hdCombo = CreateWindowEx(0, "COMBOBOX", "test",

WS\_CHILD | WS\_TABSTOP | WS\_VISIBLE | CBS\_DROPDOWNLIST,

10, 40, 175, 23,

hDialog,

IntPtr.Zero, IntPtr.Zero, IntPtr.Zero);

SendMessage(hdCombo, CB\_ADDSTRING, IntPtr.Zero, "Параметр 1");

SendMessage(hdCombo, CB\_ADDSTRING, IntPtr.Zero, "Параметр 2");

SendMessage(hdCombo, CB\_SETCURSEL, IntPtr.Zero, null);

}

else if (lParam == hAbout) // про окно

MessageBox(hWnd, "Это окно создано WinAPI", "Про окно", 0);

else if (lParam == hExit) // выход

DestroyWindow(hWnd);

else if (lParam == hdOk) // окай

{

StringBuilder sb = new StringBuilder();

GetWindowText(hdText, sb, sb.Capacity);

SetWindowText(hText, sb.ToString());

GetWindowText(hdCombo, sb, sb.Capacity);

SetWindowText(hParam, sb.ToString());

DestroyWindow(hWnd);

}

else if (lParam == hdHelp) // справка

MessageBox(hWnd, "Этот диалог создан WinAPI", "Про диалог", 0);

else if (lParam == hdCancel) // отмена

DestroyWindow(hWnd);

}

return DefWindowProc(hWnd, msg, wParam, lParam);

}

#endregion

}}

**КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Які бувають класи пріоритету процесу?
2. Якою функцією створюють новий процес?
3. Які головні дії виконує функція Windows API *CreateProcess*?
4. Яким є клас пріоритету процесу за замовчуванням?
5. У чому полягає специфіка використання класу пріоритету процесу REALTIME\_PRIORITY\_CLASS?
6. Як при створенні нового процессу вказати його пріоритет?
7. Як функція *CreateProcess* взаємодіє з підсистемами Windows?
8. Напишіть фрагмент програми для запуску прикладної програми, яка розташована за шляхом C:\Windows\notepad.exe. використайте для цього функцію *CreateProcess*.
9. Як при створенні процесу задати його пріоритет?
10. Які параметри функції *CreateProcess* є обов’язковими?
11. Як отримати дескриптор новоствореного процесу?
12. Чи можна при виклику функції *ShellExecute* задати пріоритет нового процесу?
13. Які є відмінності та спільні риси у роботі функцій *CreateProcess* та *WinExec*?
14. Які ви знаєте функції API для створення нового процесу?
15. Як програмно змінити пріоритет довільного процесу?
16. Як програмно змінити пріоритет власного процесу?
17. За допомогою якої функції отримують значення класу пріоритету процесу?
18. Яким чином можна програмно припинити виконання процесу?
19. Напишіть фрагмент програми, який запускає довільну прикладну програму й очікує на її завершення.
20. Для чого призначена функція API *OpenProcess*?
21. Як задати потрібний рівень доступу до процесу при виклику функції *OpenProcess*?
22. Для чого призначений дескриптор процесу?
23. Що таке ідентифікатор процесу?
24. Як можна отримати список ідентифікаторів запущених процесів?
25. Який системний процес має ідентифікатор 0?
26. Для чого використовують функцію *EnumProcessModules*?
27. Що таке «квант» в термінах ОС Windows?
28. У чому полягає принцип витісняю чого планування у керуванні потоками?
29. Яка частина ОС здійснює керування потоками?
30. Які є рівні пріоритету потоків?
31. Який взаємозв’язок між класами пріоритету процесу та рівнями пріоритету потоку?
32. Який рівень пріоритету потоку використовується за замовчуванням?
33. Що таке базовий пріоритет потоку?
34. Яка відмінність між базовим та поточним рівнями пріоритету?
35. Для чого Windows використовує поточне значення пріоритету?
36. У яких станах можуть перебувати потоки?
37. У чому полягає самостійне перемикання потоків?
38. Як відбувається витіснення одного потоку іншим?
39. Як відбувається перемикання потоків при завершенні кванта?
40. Що охоплює поняття контексту потоку?
41. Які переваги багатопотоковості у програмах?
42. Яка функція API призначення для створення нового потоку?
43. Як при створенні потоку задають його виконавчий код?
44. Як створити потік, не запускаючи його автоматично на виконання?
45. За допомогою яких функцій можна прочитати/ встановити пріоритет потоку?