ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

Харківський національний університет радіоелектроніки

Кваліфікаційна робота

«Метод адміністрування драйверів та служб на локальному хості під керуванням Windows»

Виконав:
Студ. гр. СПм-20-3
Шебанов Є.О.

Керівник:
к.т.н, доцент
Голубничий Д.Ю.

Мета та завдання кваліфікаційній роботи

Мета кваліфікаційної роботи :

Покращення процесу адміністрування сервісів та драйверів на локальному вузлі під керівництвом операційної системи Windows за рахунок створення програми-монітору за їх схованими від користувача операціями вводу/виводу

Науково-технічна задача, що вирішується у кваліфікаційній роботі :

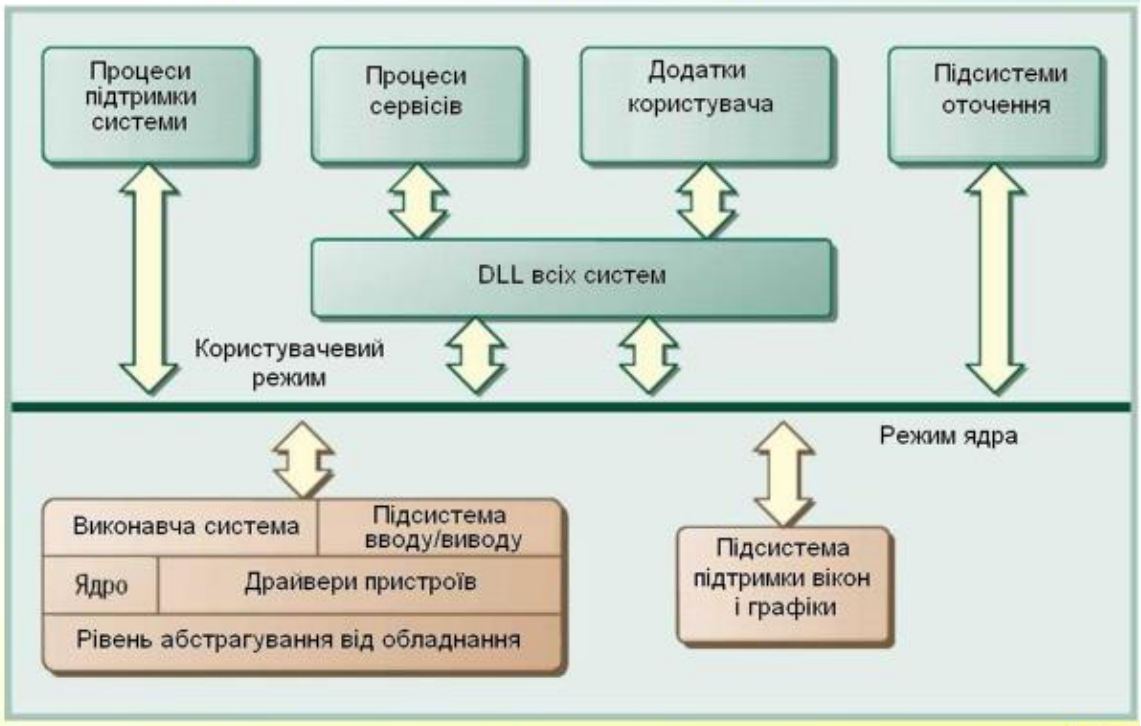
1.Провести аналіз принципів побудови підсистеми вводу/виводу Windows

2.Визначити програмні методи управління службами і драйверами

3.Запропонувати метод адміністрування службами й драйверами

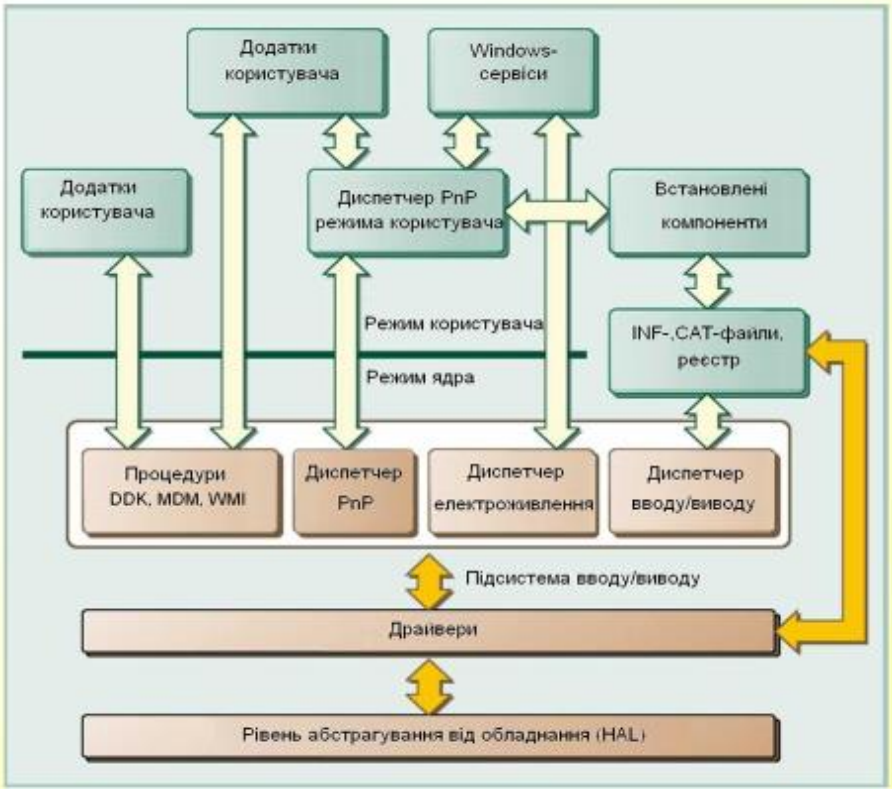
4.Побудувати програмну модель адміністрування драйверів та сервісів

Спрощена схема архітектури Windows



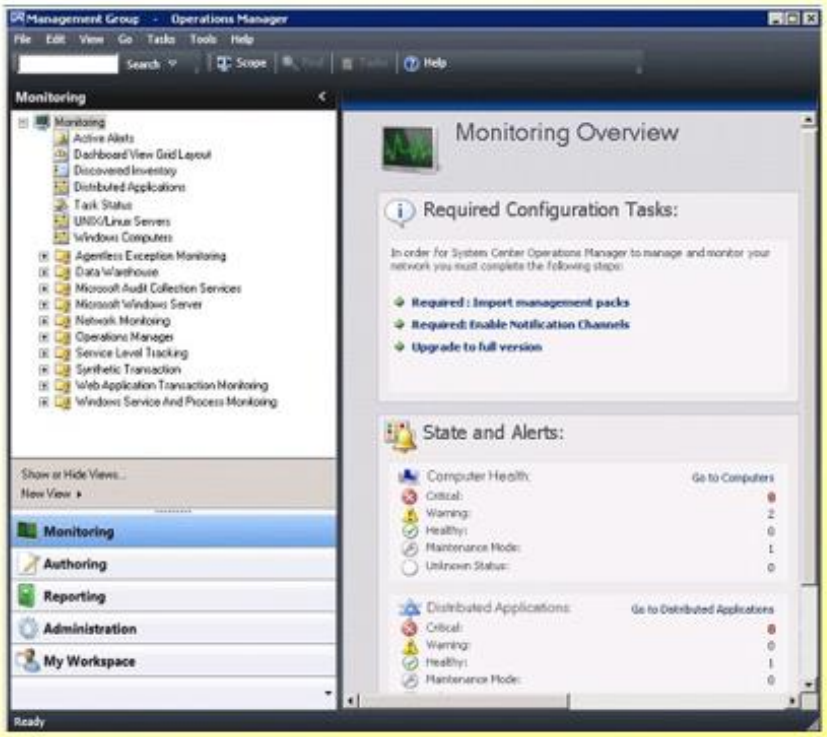
3

Компоненти підсистеми вводу/виводу



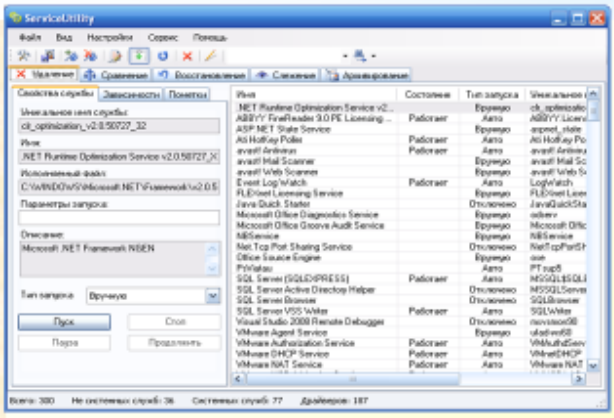
4

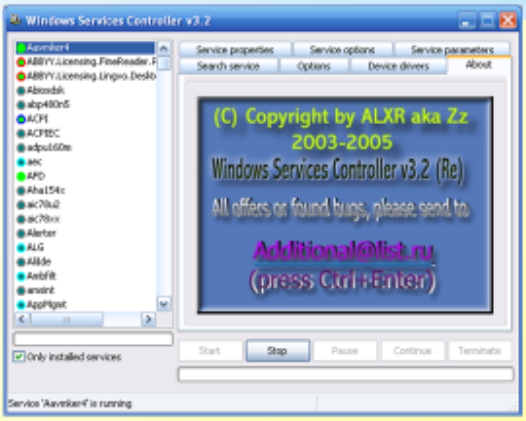
Головне вікно System Center Operations Manager



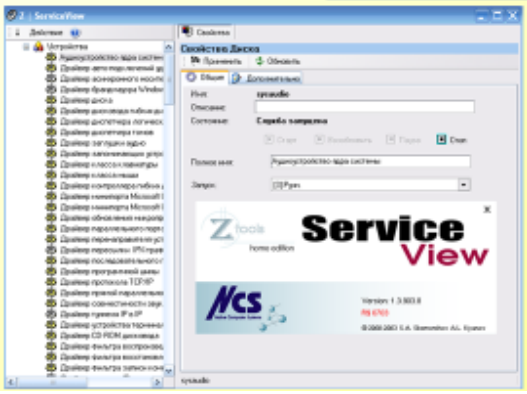
5

Програмні застосунки адміністрування сервісів





Головне вікно ServiceUtility



Головне вікно Windows Services Controller 3.2

Головне вікно ServiceView

6

Основні складові Windows-сервісів

СКЛАДОВІ WINDOWS-СЕРВІСІВ

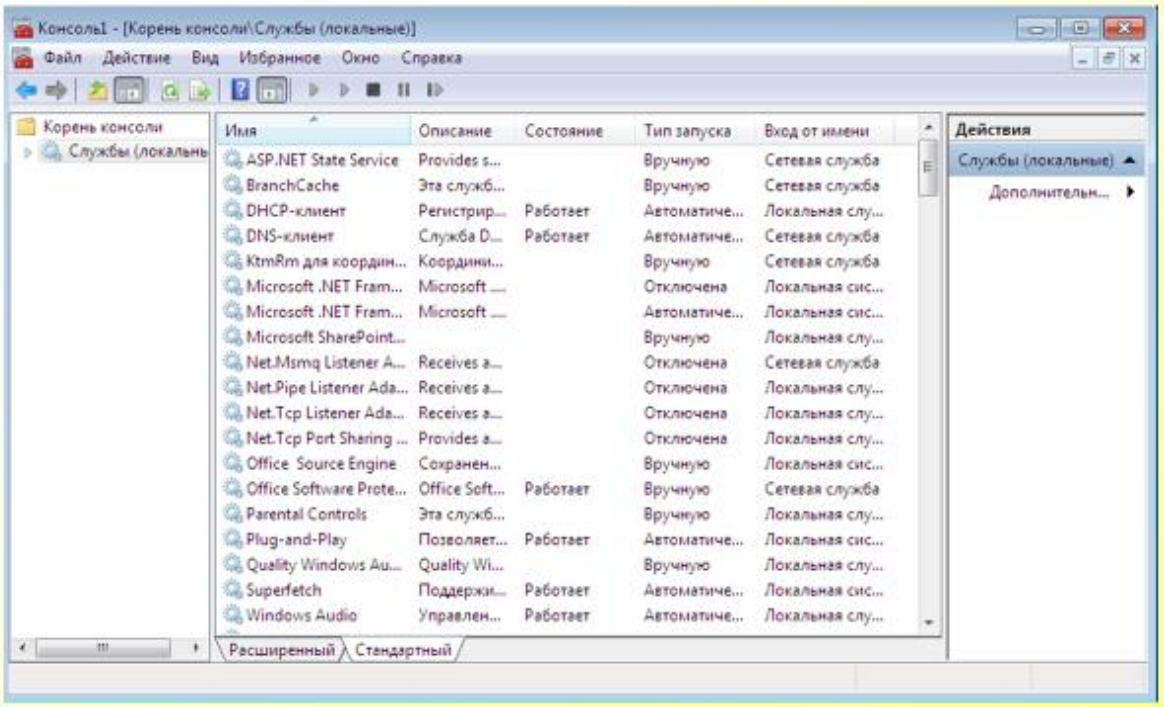
Додаток сервісу (service application)

Програма управління сервісом (service control program, **SCP**)

Диспетчер управління сервісом (service control manager, **SCM**)

7

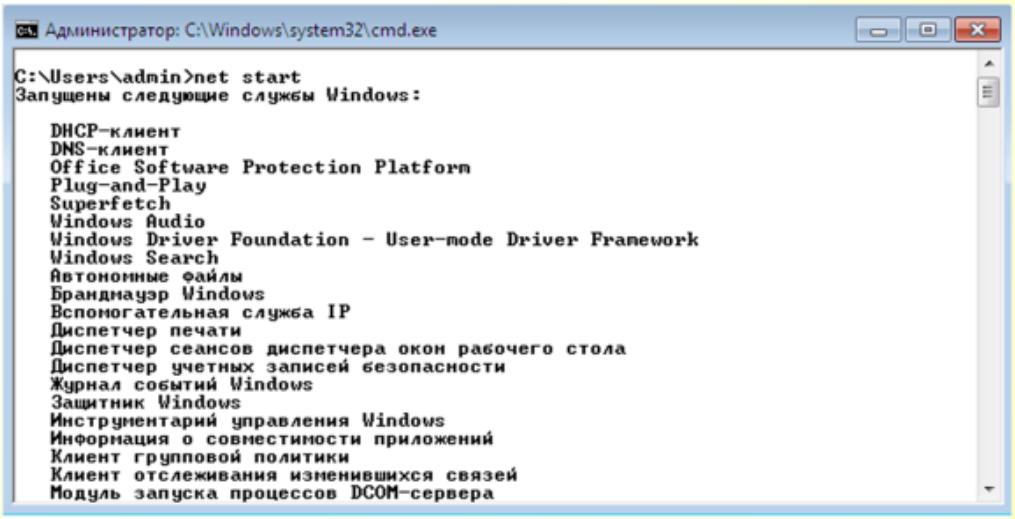
Управління службами з боку користувача



Вікно консолі MMC з відкритим оснащенням Служби

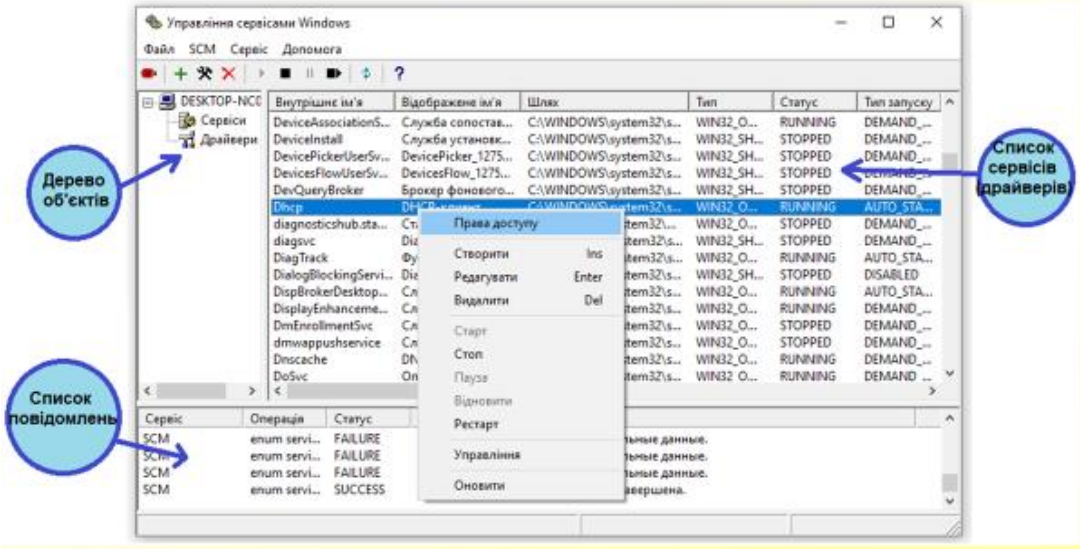
8

SCP-утиліта командного рядка Net.exe



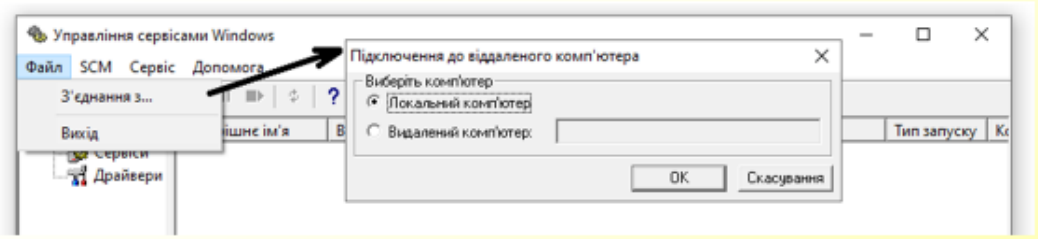
9

Головне вікно програми MonDrvSrv.exe

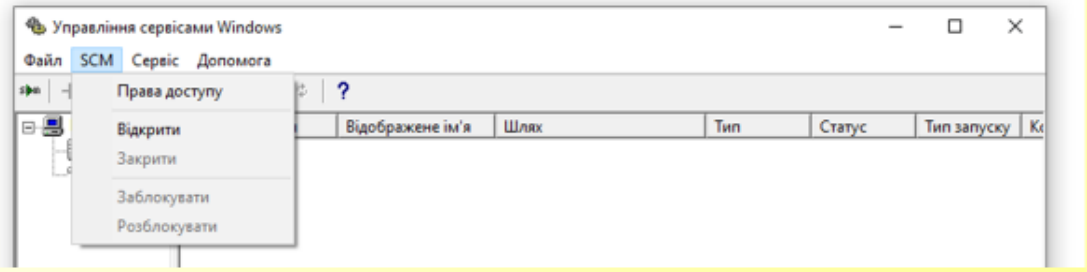


10

Адміністрування сервісів та драйверів



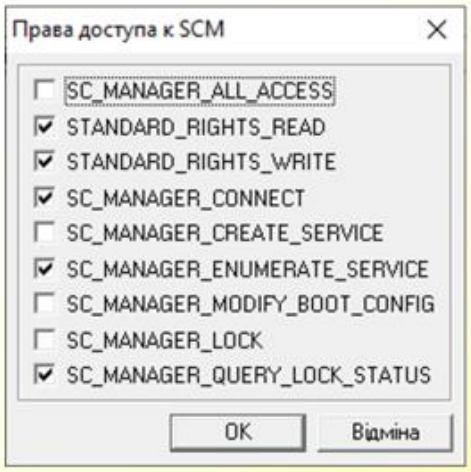
Діалогове вікно встановлення з'єднання з віддаленим комп'ютером



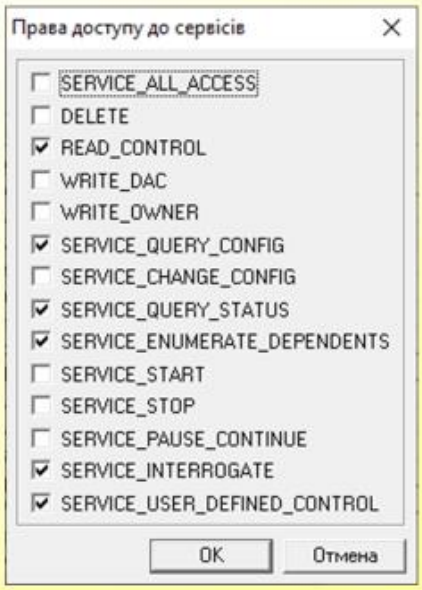
Діалогове вікно встановлення з'єднання з менеджером SCM

11

Діалогові вікна налаштувань



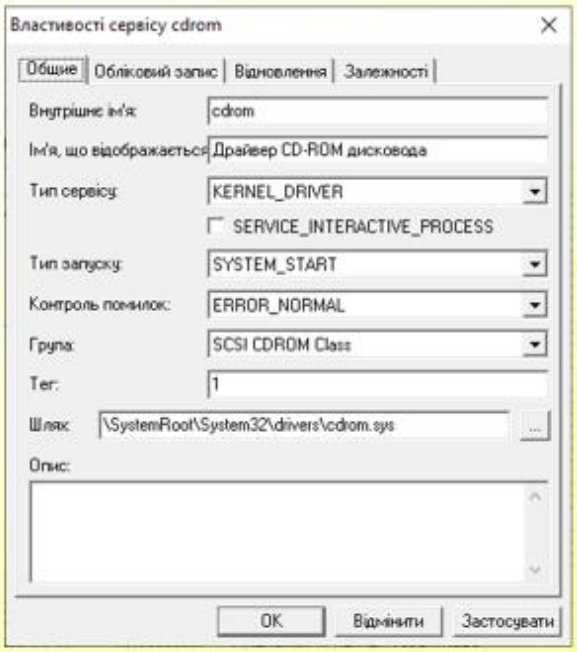
Діалогове вікно настройки прав доступу до SCM



Діалогове вікно настройки прав доступу до сервісу

12

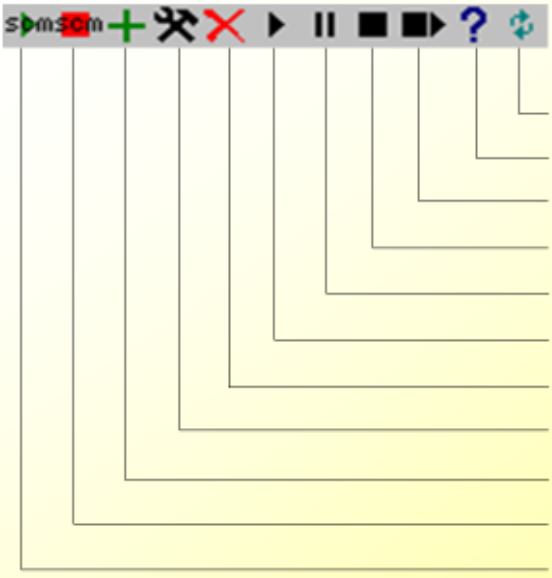
Діалогове вікно властивостей сервісу (драйверу)



Сервіс ***Драйвер***

13

Панель інструментів та призначення кнопок



14

Висновки

В результаті даної роботи були:

1.Зроблений аналіз архітектури операційної системи Windows та її підсистеми вводу/виводу.

2.Визначені основні структури даних, які пов'язані із драйверами пристроїв Windows.

3.Проведений аналіз таких програмних продуктів, як Microsoft System Center Operations Manager Enterprise, ServiceUtility, ServiceView, Windows Services Controller. Надані їх переваги та недоліки.

4.Розроблена програма дозволяє переглядати список установлених сервісів і драйверів як на локальному, так і на віддаленому комп'ютері, а також адмініструванти ними, реєструвати, змінювати параметри і зупиняти сервіс або драйвер.

Розроблений застосунок можна вдосконалити в декількох напрямках:

✓можливість запису логів дій адміністраторів;

✓динамічна індикація поточного стану працездатності сервісів та драйверів як на окремому комп'ютері, так й на всіх комп'ютерах мережі.

15

ДОДАТОК Б

Лістинги основних файлів програмної моделі

Б.1 Лістинг головного файлу проекту main.asm

```
;-------------------------------------------------------------
; Керування драйверами Windows
;
; Файл: main.asm
; Опис: Головний файл проекту. Створення вікна та обробка
повідомлень.
; Автор: Шебанов Євгеній Олександрович
;-------------------------------------------------------------

title    Main

.386
.model flat,stdcall
option casemap:none

include ..\..\Masm32\include\windows.inc
include ..\..\Masm32\include\kernel32.inc
include ..\..\Masm32\include\user32.inc
include ..\..\Masm32\include\gdi32.inc
include ..\..\Masm32\include\comctl32.inc
include ..\..\Masm32\include\comdlg32.inc
include ..\..\Masm32\include\advapi32.inc
include ..\..\Masm32\include\shell32.inc

includelib ..\..\Masm32\lib\kernel32.lib
includelib ..\..\Masm32\lib\user32.lib
includelib ..\..\Masm32\lib\gdi32.lib
includelib ..\..\Masm32\lib\comctl32.lib
includelib ..\..\Masm32\lib\comdlg32.lib
includelib ..\..\Masm32\lib\advapi32.lib
includelib ..\..\Masm32\lib\shell32.lib

include res.inc
include global.inc
include main.inc
include memory.inc
include services.inc
include servicedlg.inc
include controlservicedlg.inc
include subserventdlg.inc
include about.inc
include log.inc

; Ініціалізовані дані
```

```
.data
    szClassName db "SM_MainWindow",0
    szAppName   db "Управління сервісами Windows",0
    szMenuName  db 'MainMenu',0
    szTreeMenuName  db 'TreeViewMenu',0
    szOnCloseMessage db 'Закрийте усі дочірні вікна.', 0
    szDeleteServiceMessage db 'Ви дійсно бажаєте видалити сервіс
%s?', 0
    szMainAcceleratorTable db 'MainAccel',0
    szServAcceleratorTable db 'ServAccel',0
    szExportFilter db "Text Files (*.txt)",0,"*.txt",0,0
    szDefTxtExt db '.txt',0

    szToolBarClassName db 'ToolbarWindow32', 0
    szStatusBarClassName db 'msctls_statusbar32', 0
    szTreeViewClassName db 'SysTreeView32', 0
    szListViewClassName db 'SysListView32', 0

    sbParts dd 420, 620,  -1

    dwStartButtonId dd IDB_SERV_START
    dwControlWindowCount dd 0

    tbbuttons   TBBUTTON<0, IDB_SCM_OPEN, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SCM_OPEN>
                    TBBUTTON<0, 0, TBSTATE_ENABLED, TBSTYLE_SEP,
0, 0>
                    TBBUTTON<2, IDB_SERV_NEW, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SERV_NEW>
                    TBBUTTON<3, IDB_SERV_EDIT, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SERV_EDIT>
                    TBBUTTON<4, IDB_SERV_DELETE,
TBSTATE_ENABLED, TBSTYLE_BUTTON, 0, IDB_SERV_DELETE>
                    TBBUTTON<0, 0, TBSTATE_ENABLED,TBSTYLE_SEP,
0, 0>
                    TBBUTTON<5, IDB_SERV_START, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SERV_START>
                    TBBUTTON<7, IDB_SERV_STOP, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SERV_STOP>
                    TBBUTTON<6, IDB_SERV_PAUSE, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_SERV_PAUSE>
                    TBBUTTON<8, IDB_SERV_RESTART,
TBSTATE_ENABLED, TBSTYLE_BUTTON, 0, IDB_SERV_RESTART>
                    TBBUTTON<0, 0, TBSTATE_ENABLED, TBSTYLE_SEP,
0, 0>
                    TBBUTTON<10, IDB_REFRESH, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_REFRESH>
                    TBBUTTON<0, 0, TBSTATE_ENABLED, TBSTYLE_SEP,
0, 0>
                    TBBUTTON<9, IDB_HELP, TBSTATE_ENABLED,
TBSTYLE_BUTTON, 0, IDB_HELP>
                    TBBUTTON<1, IDB_SCM_CLOSE, TBSTATE_ENABLED
or TBSTATE_HIDDEN, TBSTYLE_BUTTON, 0, IDB_SCM_CLOSE>
```

```
                    TBBUTTON<5, IDB_SERV_RESUME,
TBSTATE_ENABLED, TBSTYLE_BUTTON, 0, IDB_SERV_RESUME>

    tvitems      TreeViewItem<0,0, 0, 0, SERVICE_WIN32 or
SERVICE_DRIVER, >
                    TreeViewItem<0,0, 1, 1, SERVICE_WIN32,
'Сервіси'>
                    TreeViewItem<0,0, 2, 2, SERVICE_DRIVER,
'Драйвери'>

    LogColumns        ColumnInfo <CT_LOG_SERV, DT_STRING,
'Сервіс', 100, Black,0>
                                ColumnInfo <CT_LOG_OP, DT_INTEGER,
'Операція', 80, Black,0>
                                ColumnInfo <CT_LOG_STATUS,
DT_INTEGER, 'Статус', 80, Black,0>
                                ColumnInfo <CT_LOG_ERRCODE,
DT_STRING, 'Код помилки', 80, Black,0>
                                ColumnInfo <CT_LOG_ERRDESC,
DT_STRING, 'Опис помилки', 400, Black,0>

    ServicesColumns ColumnInfo <CT_SERV_INAME, DT_STRING,
'Внутрішнє ім''я', 120, Black,0>
                                ColumnInfo <CT_SERV_DNAME,
DT_STRING,  'Відображене ім''я', 120, Black,0>
                                ColumnInfo <CT_SERV_DNAME,
DT_STRING,  'Шлях', 160, Black,0>
                                ColumnInfo <CT_SERV_TYPE,
DT_INTEGER, 'Тип', 80, Black,0>
                                ColumnInfo <CT_SERV_START_TYPE,
DT_INTEGER, 'Статус', 80, Black,0>
                                ColumnInfo <CT_SERV_START_TYPE,
DT_INTEGER, 'Тип запуску', 80, Black,0>
                                ColumnInfo <CT_SERV_ERROR_TYPE,
DT_INTEGER, 'Контроль помилок', 105, Black,0>
                                ColumnInfo <CT_SERV_LOGON,
DT_STRING, 'Обліковий запис', 100, Black,0>
                                ColumnInfo <CT_SERV_DESCRIPTION,
DT_STRING, 'Опис', 400, Black,0>

    tbinfo  ToolBarInfo <szAppName, NULL, IDC_BUTTONS,
BUTTONCOUNT, TB_SIZE, TB_SIZE, BUTTONCOUNT, TB_SIZE, TB_SIZE,
tbbuttons>
    tvinfo  ToolBarInfo <NULL, IDC_TREEVIEWITEMS,
IDC_TREEVIEWMASK, TVI_COUNT, TVI_SIZE, TVI_SIZE, TVI_COUNT,
tvitems>

    dwSCMAccess dd STANDARD_RIGHTS_READ OR SC_MANAGER_CONNECT OR
SC_MANAGER_ENUMERATE_SERVICE OR SC_MANAGER_QUERY_LOCK_STATUS
    dwServAccess dd READ_CONTROL OR SERVICE_ENUMERATE_DEPENDENTS
OR SERVICE_INTERROGATE OR SERVICE_QUERY_CONFIG OR
SERVICE_QUERY_STATUS OR SERVICE_USER_DEFINED_CONTROL
    dwServiceEnumType dd SERVICE_WIN32 or SERVICE_DRIVER
```

```
    szOpen  db 'open',0
    szFmtS db "%s",0
    szFmtD db "%d",0
    szFmtX db "%#08lx",0

; Неініціалізовані дані
.data?
    hInstance HINSTANCE ?
    lpCommandLine LPSTR ?

    hMainWnd HWND ?
    hMainMenu HMENU ?
    hMainToolBar HWND ?
    hMainStatus HWND ?
    hMainTreeView HWND ?
    hServiceList HWND ?
    hLogList HWND ?

    hServPopupMenu HMENU ?
    hTreePopupMenu HMENU ?

    hMainAccelerators HANDLE ?
    hServAccelerators HANDLE ?
    hCurrentAccelerators HANDLE ?

    hSCManager HANDLE ?
    sclLock DWORD ?

    cnCompData ComputorNameData<?>

    hNormalCursor HCURSOR ?
    hVertSplitCursor HCURSOR ?
    hHortSplitCursor HCURSOR ?

    ClientRect RECT <?>
    uTopHeight dd ?
    uBottomHeight dd ?
    uLeftWidth dd ?
    uRightWidth dd ?

    VertSplitter RECT <?>
    HortSplitter RECT <?>
    DeltaX dd ?
    DeltaY dd ?
    IsHCapture db ?
    IsVCapture db ?

    ofsExportPath OPENFILENAME <>
    szTmpBuffer db TMP_BUF_SIZE dup(?)
    szTmpBuffer2 db TMP_BUF_SIZE dup(?)

    lpMsgBuf dd ?
```

```
; Код
.code

begin:
    invoke GetModuleHandle, NULL
    mov hInstance, eax
    invoke GetCommandLine
    mov lpCommandLine, eax

    invoke WinMain, hInstance,NULL,lpCommandLine,SW_SHOWDEFAULT
    invoke ExitProcess, eax
    invoke InitCommonControls

;-------------------------------------------------------------
; WinMain
; - hInst - Хінстанс програми
; - hPrevInst - Хінстанс попередньої програми(не
використовується)
; - lpCmdLine - командний рядок
; - nCmdShow - прапор показу вікна
;-------------------------------------------------------------
WinMain proc hInst:HINSTANCE, hPrevInst:HINSTANCE,
lpCmdLine:LPSTR, nCmdShow:DWORD
    LOCAL wc:WNDCLASSEX
    LOCAL msg:MSG

    ; ініціалізація класу вікна
    mov wc.cbSize,SIZEOF WNDCLASSEX
    mov wc.style,CS_HREDRAW or CS_VREDRAW
    mov wc.lpfnWndProc, offset MainWndProc
    mov wc.cbClsExtra,NULL
    mov wc.cbWndExtra,NULL
    push hInst
    pop wc.hInstance
    mov wc.hbrBackground,COLOR_BTNFACE+1
    mov wc.lpszMenuName, offset szMenuName
    mov wc.lpszClassName,offset szClassName
    invoke LoadIcon, hInst, IDC_MAINICON
    mov wc.hIcon,eax
    mov wc.hIconSm,eax
    invoke LoadCursor,NULL,IDC_ARROW
    mov wc.hCursor,eax
    mov hNormalCursor, eax

    ; реєстрація класу вікна
    invoke RegisterClassEx, addr wc

    ; створення вікна
    invoke CreateWindowEx, NULL, addr szClassName,addr
szAppName, WS_OVERLAPPEDWINDOW,\
           CW_USEDEFAULT, CW_USEDEFAULT,800, 600,\
           NULL, NULL, hInst,NULL
```

```
    mov hMainWnd,eax
    invoke ShowWindow, eax,nCmdShow
    invoke UpdateWindow, eax

    ; акселератори
    invoke LoadAccelerators, hInst, addr szMainAcceleratorTable
    mov hMainAccelerators, eax
    mov hCurrentAccelerators, eax
    invoke LoadAccelerators, hInst, addr szServAcceleratorTable
    mov hServAccelerators, eax

    ; цикл обробки повідомлень
    .while TRUE
        invoke GetMessage , addr msg ,NULL ,0,0
        .break .if (!eax)
        invoke TranslateAccelerator,hMainWnd,
hCurrentAccelerators, addr msg
        .if (eax == 0)
            invoke TranslateMessage ,addr msg
            invoke DispatchMessage ,addr msg
        .endif
    .endw

    mov eax,msg.wParam
    ret
WinMain endp

;-------------------------------------------------------------
; MainWndProc - Віконна процедура головного вікна програми
; - hWnd - хендл вікна
; - uMsg - повідомлення
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
MainWndProc proc hWnd:HWND, uMsg:UINT, wParam:WPARAM,
lParam:LPARAM
    .if uMsg==WM_DESTROY
        invoke FreeResources
        invoke PostQuitMessage,NULL
    .elseif uMsg==WM_CLOSE
        invoke Exit, hWnd
    .elseif uMsg==WM_CREATE
        invoke OnCreate, hWnd
    .elseif uMsg==WM_GETMINMAXINFO
        invoke GetMinMaxInfo, lParam
    .elseif uMsg==WM_SIZE
        .if (wParam != SIZE_MINIMIZED)
            invoke OnSize, hWnd
        .endif
    .elseif uMsg==WM_LBUTTONDOWN
        invoke OnLButtonDown, hWnd,wParam,lParam
    .elseif uMsg==WM_LBUTTONUP
        invoke OnLButtonUp, hWnd,wParam,lParam
```

```
    .elseif uMsg==WM_MOUSEMOVE
        invoke OnMouseMove, hWnd,wParam,lParam
    .elseif uMsg ==WM_NOTIFY
        invoke OnNotify, hWnd, wParam, lParam
    .elseif uMsg==WM_COMMAND
        mov eax, wParam
        and eax, 0ffffh
        .if  (eax == IDM_CONNECTTO)
            invoke SCMConnectTo, hWnd
        .elseif (eax == IDM_EXPORT)
            invoke ExportDataToFile, hServiceList
        .elseif (eax == IDM_EXIT)
            invoke Exit, hWnd
        .elseif (eax == IDM_SCM_ACCESS)
            invoke OpenSCMAccess, hWnd
        .elseif (eax == IDM_SCM_OPEN) || (eax == IDB_SCM_OPEN)
            invoke OpenSCMConnection
        .elseif (eax == IDM_SCM_CLOSE) || (eax == IDB_SCM_CLOSE)
            invoke CloseSCMConnection
        .elseif (eax == IDM_SCM_LOCK)
            invoke LockSCM
        .elseif (eax == IDM_SCM_UNLOCK)
            invoke UnLockSCM
        .elseif (eax == IDM_SERV_ACCESS)
            invoke OpenServAccess, hWnd
        .elseif (eax == IDM_SERV_SECURITY)
            invoke OpenServSecurity, hWnd
        .elseif (eax == IDM_SERV_NEW) || (eax == IDB_SERV_NEW)
            invoke AddServRecord, hServiceList
        .elseif (eax == IDM_SERV_EDIT) || (eax == IDB_SERV_EDIT)
            invoke EditServRecord, hServiceList
        .elseif (eax == IDM_SERV_DELETE) || (eax ==
IDB_SERV_DELETE)
            invoke DeleteServRecord, hServiceList
        .elseif (eax == IDM_SERV_START) || (eax ==
IDB_SERV_START)
            invoke ServControl, hServiceList,
ServiceStartOperation
        .elseif (eax == IDM_SERV_STOP) || (eax == IDB_SERV_STOP)
            invoke ServControl, hServiceList,
ServiceStopOperation
        .elseif (eax == IDM_SERV_PAUSE) || (eax ==
IDB_SERV_PAUSE)
            invoke ServControl, hServiceList,
ServicePauseOperation
        .elseif (eax == IDM_SERV_RESUME) || (eax ==
IDB_SERV_RESUME)
            invoke ServControl, hServiceList,
ServiceResumeOperation
        .elseif (eax == IDM_SERV_RESTART) || (eax ==
IDB_SERV_RESTART)
            invoke ServControl, hServiceList,
ServiceRestartOperation
```

```
        .elseif (eax == IDM_SERV_IOCONTROL)
            invoke ServControlDlg, hWnd, hServiceList
        .elseif (eax == IDM_REFRESH) || (eax == IDB_REFRESH)
            invoke RefreshData, hServiceList
        .elseif (eax == IDM_ABOUT) || (eax == IDB_HELP)
            invoke OpenAboutDlg, hWnd
        .endif
    .else
        invoke DefWindowProc,hWnd,uMsg,wParam,lParam
        ret
    .endif

    xor eax,eax
    ret
MainWndProc endp

;-------------------------------------------------------------
; OnCreate - Обробка події WM_CREATE головного вікна
; - hWnd - хендл вікна
;-------------------------------------------------------------
OnCreate proc hWnd:HWND
    LOCAL ChildRect: RECT
    LOCAL WindowWidth: DWORD
    LOCAL WindowHeight: DWORD
    LOCAL col: LV_COLUMN
    LOCAL nSize: DWORD

    ; ініціалізувати дані
    mov IsHCapture, FALSE
    mov IsVCapture, FALSE

    xor eax, eax
    mov hSCManager, eax
    mov sclLock, eax

    invoke malloc, dwStatusInfoSize
    mov pStatusInfo, eax
    invoke malloc, dwConfigInfoSize
    mov pConfigInfo, eax
    invoke malloc, dwDescriptionInfoSize
    mov pDescriptionInfo, eax
    invoke malloc, dwFailureInfoSize
    mov pFailureInfo, eax
    invoke malloc, dwGroupInfoSize
    mov pGroupInfo, eax

    mov cnCompData.dwCompFlag, CND_LOCALCOMP
    mov cnCompData.dwCompNameSize, 256
    invoke malloc, cnCompData.dwCompNameSize
    mov cnCompData.lpCompName, eax

    mov nSize, sizeof tvitems.ItemString
    invoke GetComputerName, addr tvitems.ItemString, addr nSize
```

```
    ; завантажити ресурси
    invoke LoadCursor, hInstance, IDC_SPLITV
    mov hVertSplitCursor, eax
    invoke LoadCursor, hInstance, IDC_SPLITH
    mov hHortSplitCursor, eax

    invoke GetMenu, hWnd
    mov hMainMenu, eax

    ;создать  StatusBar
    invoke CreateWindowEx, WS_EX_OVERLAPPEDWINDOW, addr
szStatusBarClassName, 0, WS_CHILD OR WS_VISIBLE OR
SBS_SIZEGRIP,\
                              0, 0, 0, 0, hWnd,0,hInstance,0

    mov hMainStatus, eax
    invoke SetWindowLong, hMainStatus, GWL_ID, IDC_MAINSTATUS
    invoke SendMessage, hMainStatus,SB_SETPARTS, 3, addr sbParts

    ;створити ToolBar
    invoke CreateToolBar, hWnd, hInstance, addr tbinfo
    mov hMainToolBar, eax

    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_NEW, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_EDIT, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_DELETE, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_START, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_STOP, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_PAUSE, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESUME, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESTART, FALSE
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_REFRESH, FALSE

    ;розраховуємо область для дочірніх вікон
    invoke GetClientRect, hWnd, addr ClientRect
    invoke GetClientRect, hMainToolBar, addr ChildRect
    push ChildRect.bottom
    pop ClientRect.top
    add ClientRect.top, 2
    invoke GetClientRect, hMainStatus, addr ChildRect
    mov eax, ChildRect.bottom
    sub eax, ChildRect.top
    sub ClientRect.bottom, eax
```

```
push ClientRect.bottom
pop WindowHeight
mov eax, ClientRect.top
sub WindowHeight, eax
push ClientRect.right
pop WindowWidth
mov eax, ClientRect.left
sub WindowWidth, eax

push TREE_WIDTH
pop uLeftWidth
push LOG_HEIGHT
pop uBottomHeight

mov eax, WindowWidth
sub eax, SPLITTER_SIZE
.if (! SIGN?)
    .if (WindowWidth > 2*MIN_SIZE)
        .if (uLeftWidth < MIN_SIZE)
            mov uLeftWidth, MIN_SIZE
        .endif
        sub eax, uLeftWidth
        .if (SIGN?) || (eax < MIN_SIZE)
            mov uRightWidth, MIN_SIZE
            mov eax, WindowWidth
            sub eax, SPLITTER_SIZE
            sub eax, uRightWidth
            mov uLeftWidth, eax
        .else
            mov uRightWidth, eax
        .endif
    .elseif (WindowWidth > MIN_SIZE)
        mov uLeftWidth, MIN_SIZE
        sub eax, uLeftWidth
        mov uRightWidth, eax
    .else
        push WindowWidth
        pop uLeftWidth
        xor eax, eax
        mov uRightWidth, eax
    .endif
.elseif
    push WindowWidth
    pop uLeftWidth
    xor eax, eax
    mov uRightWidth, eax
.endif

mov eax, WindowHeight
sub eax, SPLITTER_SIZE
.if (! SIGN?)
    .if (WindowHeight > 2*MIN_SIZE)
```

```
            .if (uBottomHeight < MIN_SIZE)
                mov uBottomHeight, MIN_SIZE
            .endif
            sub eax, uBottomHeight
            .if (SIGN?) || (eax < MIN_SIZE)
                mov uTopHeight, MIN_SIZE
                mov eax, WindowHeight
                sub eax, SPLITTER_SIZE
                sub eax, uTopHeight
                mov uBottomHeight, eax
            .else
                mov uTopHeight, eax
            .endif
        .elseif (WindowHeight > MIN_SIZE)
            mov uBottomHeight, MIN_SIZE
            sub eax, uBottomHeight
            mov uTopHeight, eax
        .else
            push WindowHeight
            pop uBottomHeight
            xor eax, eax
            mov uTopHeight, eax
        .endif
    .elseif
        push WindowHeight
        pop uBottomHeight
        xor eax, eax
        mov uTopHeight, eax
    .endif

    ; розрахунок сплитерів
    push ClientRect.left
    pop HortSplitter.left
    push ClientRect.right
    pop HortSplitter.right
    mov eax, ClientRect.top
    add eax, uTopHeight
    mov HortSplitter.top, eax
    add eax, SPLITTER_SIZE
    mov HortSplitter.bottom, eax

    mov eax, ClientRect.top
    mov VertSplitter.top, eax
    add eax, uTopHeight
    mov VertSplitter.bottom, eax
    mov eax, ClientRect.left
    add eax, uLeftWidth
    mov VertSplitter.left, eax
    add eax, SPLITTER_SIZE
    mov VertSplitter.right, eax

    ;створити дочірні вікна
    invoke CreateTreeView, hWnd, hInstance, addr tvinfo
```

```
    mov hMainTreeView, eax

    mov eax, ClientRect.bottom
    sub eax, uBottomHeight
    invoke CreateWindowEx, WS_EX_CLIENTEDGE, addr
szListViewClassName, NULL,  WS_CHILD or WS_VISIBLE or LVS_REPORT
or LVS_SINGLESEL or LVS_NOSORTHEADER or LVS_SHOWSELALWAYS, \
                  ClientRect.left, eax, WindowWidth,
uBottomHeight, hWnd, 0, hInstance, NULL
    mov hLogList, eax
    invoke SendMessage, eax, LVM_SETEXTENDEDLISTVIEWSTYLE,
LVS_EX_FULLROWSELECT, LVS_EX_FULLROWSELECT

    mov eax, ClientRect.left
    add eax, uLeftWidth
    add eax, SPLITTER_SIZE
    invoke CreateWindowEx, WS_EX_CLIENTEDGE, addr
szListViewClassName, NULL,  WS_CHILD or WS_VISIBLE or LVS_REPORT
or LVS_SINGLESEL or LVS_SHOWSELALWAYS or LVS_SORTASCENDING,\
                  eax, ClientRect.top, uRightWidth,
uTopHeight, hWnd, 0, hInstance, NULL
    mov hServiceList, eax
    invoke SendMessage, eax, LVM_SETEXTENDEDLISTVIEWSTYLE,
LVS_EX_FULLROWSELECT, LVS_EX_FULLROWSELECT

    ; створити та заповнити колонки
    pushad

    mov col.imask, LVCF_FMT OR LVCF_SUBITEM OR LVCF_TEXT OR
LVCF_WIDTH
    mov col.fmt, LVCFMT_LEFT
    xor eax, eax
    mov ecx, 6
    lea edi, col.lx
    ;mov edi, offset col.lx
    rep stosd

    xor esi, esi
    mov edi,SEVR_COLUMN_COUNT
    mov ebx, offset ServicesColumns
    .while (esi < edi)
        assume ebx:ptr ColumnInfo
        lea eax, [ebx].ColumnName
        mov col.pszText, eax
        mov col.cchTextMax, sizeof ColumnInfo.ColumnName
        mov col.iSubItem, esi
        push [ebx].ColumnWidth
        pop col.lx

        invoke SendMessage, hServiceList, LVM_INSERTCOLUMN, esi,
addr col

        add ebx, sizeof ColumnInfo
```

```
        inc esi
        assume ebx: nothing
    .endw

    xor esi, esi
    mov edi, LOG_COLUMN_COUNT
    mov ebx, offset LogColumns
    .while (esi < edi)
        assume ebx:ptr ColumnInfo
        lea eax, [ebx].ColumnName
        mov col.pszText, eax
        mov col.cchTextMax, sizeof ColumnInfo.ColumnName
        mov col.iSubItem, esi
        push [ebx].ColumnWidth
        pop col.lx

        invoke SendMessage, hLogList, LVM_INSERTCOLUMN, esi,
addr col

        add ebx, sizeof ColumnInfo
        inc esi
        assume ebx: nothing
    .endw

    ; попап меню
    invoke GetMenu, hWnd
    invoke GetSubMenu, eax, 2
    mov hServPopupMenu, eax

    invoke SetWindowLong, hServiceList, GWL_WNDPROC,
ServiceListProc
    invoke SetWindowLong, hServiceList, GWL_USERDATA, eax

    popad
    ret
OnCreate endp

;-------------------------------------------------------------
; OnSize - Обробка події WM_SIZE головного вікна
; - hWnd - хендл вікна
;-------------------------------------------------------------
OnSize proc hWnd:HWND
    LOCAL ChildRect: RECT
    LOCAL WindowWidth: DWORD
    LOCAL WindowHeight: DWORD

    invoke MoveWindow, hMainStatus,0,0,0,0,TRUE
    invoke SendMessage, hMainToolBar,TB_AUTOSIZE,0,0

    ;розраховуємо область для дочірніх вікон
    invoke GetClientRect, hWnd, addr ClientRect
    invoke GetClientRect, hMainToolBar, addr ChildRect
    push ChildRect.bottom
```

```
pop ClientRect.top
add ClientRect.top, 2
invoke GetClientRect, hMainStatus, addr ChildRect
mov eax, ChildRect.bottom
sub eax, ChildRect.top
sub ClientRect.bottom, eax
mov eax, ClientRect.top
.if (ClientRect.bottom < eax)
    mov ClientRect.bottom, eax
.endif

push ClientRect.bottom
pop WindowHeight
mov eax, ClientRect.top
sub WindowHeight, eax
push ClientRect.right
pop WindowWidth
mov eax, ClientRect.left
sub WindowWidth, eax

mov eax, WindowWidth
sub eax, SPLITTER_SIZE
.if (! SIGN?)
    .if (WindowWidth > 2*MIN_SIZE)
        .if (uLeftWidth < MIN_SIZE)
            mov uLeftWidth, MIN_SIZE
        .endif
        sub eax, uLeftWidth
        .if (SIGN?) || (eax < MIN_SIZE)
            mov uRightWidth, MIN_SIZE
            mov eax, WindowWidth
            sub eax, SPLITTER_SIZE
            sub eax, uRightWidth
            mov uLeftWidth, eax
        .else
            mov uRightWidth, eax
        .endif
    .elseif (WindowWidth > MIN_SIZE)
        mov uLeftWidth, MIN_SIZE
        sub eax, uLeftWidth
        mov uRightWidth, eax
    .else
        push WindowWidth
        pop uLeftWidth
        xor eax, eax
        mov uRightWidth, eax
    .endif
.elseif
    push WindowWidth
    pop uLeftWidth
    xor eax, eax
    mov uRightWidth, eax
.endif
```

```
mov eax, WindowHeight
sub eax, SPLITTER_SIZE
.if (! SIGN?)
    .if (WindowHeight > 2*MIN_SIZE)
        .if (uBottomHeight < MIN_SIZE)
            mov uBottomHeight, MIN_SIZE
        .endif
        sub eax, uBottomHeight
        .if (SIGN?) || (eax < MIN_SIZE)
            mov uTopHeight, MIN_SIZE
            mov eax, WindowHeight
            sub eax, SPLITTER_SIZE
            sub eax, uTopHeight
            mov uBottomHeight, eax
        .else
            mov uTopHeight, eax
        .endif
    .elseif (WindowHeight > MIN_SIZE)
        mov uBottomHeight, MIN_SIZE
        sub eax, uBottomHeight
        mov uTopHeight, eax
    .else
        push WindowHeight
        pop uBottomHeight
        xor eax, eax
        mov uTopHeight, eax
    .endif
.elseif
    push WindowHeight
    pop uBottomHeight
    xor eax, eax
    mov uTopHeight, eax
.endif

; розрахунок сплитерів
push ClientRect.left
pop HortSplitter.left
push ClientRect.right
pop HortSplitter.right
mov eax, ClientRect.top
add eax, uTopHeight
mov HortSplitter.top, eax
add eax, SPLITTER_SIZE
mov HortSplitter.bottom, eax

mov eax, ClientRect.top
mov VertSplitter.top, eax
add eax, uTopHeight
mov VertSplitter.bottom, eax
mov eax, ClientRect.left
add eax, uLeftWidth
mov VertSplitter.left, eax
```

```
    add eax, SPLITTER_SIZE
    mov VertSplitter.right, eax

    ; змінити розміри вікон
    invoke MoveWindow, hMainTreeView, ClientRect.left,
ClientRect.top, uLeftWidth, uTopHeight, TRUE

    mov eax, ClientRect.bottom
    sub eax, uBottomHeight
    invoke MoveWindow, hLogList, ClientRect.left, eax,
WindowWidth, uBottomHeight, TRUE

    mov eax, ClientRect.left
    add eax, uLeftWidth
    add eax, SPLITTER_SIZE
    invoke MoveWindow, hServiceList, eax, ClientRect.top,
uRightWidth, uTopHeight, TRUE

    ret
OnSize endp

;-------------------------------------------------------------
; GetMinMaxInfo - заповнити структуру мінімальних та
максимальних розмірів вікна
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
GetMinMaxInfo proc lParam:LPARAM
    push ebx

    mov ebx, lParam
    mov (MINMAXINFO ptr [ebx]).ptMinTrackSize.x, MIN_WIDTH
    mov (MINMAXINFO ptr [ebx]).ptMinTrackSize.y, MIN_HEIGHT

    pop ebx
    ret
GetMinMaxInfo endp


;-------------------------------------------------------------
; OnLButtonDown - Обробник натискання лівої кнопки миші
; - hWnd - хендл вікна
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
OnLButtonDown proc hWnd:HWND, wParam:WPARAM, lParam:LPARAM
    pushad
    mov eax, lParam
    mov edx, eax
    and eax, 0ffffh
    shr edx, 16

    .if (eax >= HortSplitter.left && eax <= HortSplitter.right)
&& (edx >= HortSplitter.top && edx <= HortSplitter.bottom)
```

```
; горизонтальный   сплитер
        sub edx, HortSplitter.top
        mov DeltaY, edx

        mov IsHCapture, TRUE
        invoke SetCursor, hHortSplitCursor
        invoke SetCapture, hWnd
    .elseif (eax >= VertSplitter.left && eax <=
VertSplitter.right) && (edx >= VertSplitter.top && edx <=
VertSplitter.bottom) ; вертикальный   сплитер
        sub eax, VertSplitter.left
        mov DeltaX, eax

        mov IsVCapture, TRUE
        invoke SetCursor, hVertSplitCursor
        invoke SetCapture, hWnd
    .endif

    popad
    ret
OnLButtonDown endp

;-------------------------------------------------------------
; OnLButtonDown - Обробник натискання лівої кнопки миші
; - hWnd - хендл вікна
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
OnLButtonUp proc hWnd:HWND, wParam:WPARAM, lParam:LPARAM
    mov IsHCapture, FALSE
    mov IsVCapture, FALSE
    invoke ReleaseCapture
    ret
OnLButtonUp endp

;-------------------------------------------------------------
; OnMouseMove - Обробник переміщення миші(робота сплітера)
; - hWnd - хендл вікна
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
OnMouseMove proc hWnd:HWND, wParam:WPARAM, lParam:LPARAM
    pushad
    mov eax, lParam
    mov edx, eax
    and eax, 0ffffh
    shr edx, 16

    .if (IsHCapture == TRUE)
        sub edx, DeltaY
        mov esi, ClientRect.top
        add esi, MIN_SIZE
        mov edi, ClientRect.bottom
```

```
        sub edi, MIN_SIZE
        sub edi, SPLITTER_SIZE
        .if (edx > esi && edx < edi)
            push edx
            mov uTopHeight, edx
            mov edx, ClientRect.top
            sub uTopHeight, edx
            pop edx
            add edx, SPLITTER_SIZE
            push ClientRect.bottom
            pop uBottomHeight
            sub uBottomHeight, edx

            ; спліттери
            mov eax, ClientRect.top
            add eax, uTopHeight
            mov HortSplitter.top, eax
            add eax, SPLITTER_SIZE
            mov HortSplitter.bottom, eax

            mov eax, ClientRect.top
            mov VertSplitter.top, eax
            add eax, uTopHeight
            mov VertSplitter.bottom, eax

            ; зміна розміру вікон
            invoke MoveWindow, hMainTreeView, ClientRect.left,
ClientRect.top, uLeftWidth, uTopHeight, TRUE

            mov eax, ClientRect.bottom
            sub eax, uBottomHeight
            mov edx, ClientRect.right
            sub edx, ClientRect.left
            invoke MoveWindow, hLogList, ClientRect.left, eax,
edx, uBottomHeight, TRUE

            mov eax, ClientRect.left
            add eax, uLeftWidth
            add eax, SPLITTER_SIZE
            invoke MoveWindow, hServiceList, eax,
ClientRect.top, uRightWidth, uTopHeight, TRUE
        .endif
    .elseif (IsVCapture == TRUE)
        sub eax, DeltaX
        mov esi, ClientRect.left
        add esi, MIN_SIZE
        mov edi, ClientRect.right
        sub edi, MIN_SIZE
        sub edi, SPLITTER_SIZE
        .if (eax > esi && eax < edi)
            push eax
            mov uLeftWidth, eax
            mov eax, ClientRect.left
```

```
            sub uLeftWidth, eax
            pop eax
            add eax, SPLITTER_SIZE
            push ClientRect.right
            pop uRightWidth
            sub uRightWidth, eax

            ; спліттери
            mov eax, ClientRect.left
            add eax, uLeftWidth
            mov VertSplitter.left, eax
            add eax, SPLITTER_SIZE
            mov VertSplitter.right, eax

            ; зміна розміру вікон
            invoke MoveWindow, hMainTreeView, ClientRect.left,
ClientRect.top, uLeftWidth, uTopHeight, TRUE

            mov eax, ClientRect.left
            add eax, uLeftWidth
            add eax, SPLITTER_SIZE
            invoke MoveWindow, hServiceList, eax,
ClientRect.top, uRightWidth, uTopHeight, TRUE
        .endif
    .else
        .if (eax >= HortSplitter.left && eax <=
HortSplitter.right) && (edx >= HortSplitter.top && edx <=
HortSplitter.bottom)       ; горизонтальный   сплитер
            invoke SetCursor, hHortSplitCursor
        .elseif (eax >= VertSplitter.left && eax <=
VertSplitter.right) && (edx >= VertSplitter.top && edx <=
VertSplitter.bottom) ; вертикальный   сплитер
            invoke SetCursor, hVertSplitCursor
        .endif
    .endif

    popad

    ret
OnMouseMove endp

;-------------------------------------------------------------
; OnNotify – повідомлення від дочірніх контролів
; - hWnd - хендл вікна
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
OnNotify proc hWnd:HWND, wParam:WPARAM, lParam:LPARAM
    LOCAL pItem: DWORD
    LOCAL item:LV_ITEM

    push ebx
```

```
    mov ebx, lParam
    assume ebx: ptr NMHDR
    mov eax, [ebx].hwndFrom
    .if (eax == hMainTreeView)
        .if ([ebx].code == TVN_SELCHANGED)
            assume ebx: ptr NMTREEVIEW
            mov eax, [ebx].itemNew.lParam
            mov dwServiceEnumType, eax
            invoke FillServicesList, hServiceList, hLogList,
hSCManager, eax, SERVICE_STATE_ALL
        .endif
    .elseif (eax == hServiceList)
        assume ebx: ptr NMHDR
        .if ([ebx].code == LVN_ITEMCHANGED)
            assume ebx: ptr NMLISTVIEW
            .if ([ebx].uChanged == LVIF_STATE) &&
([ebx].uNewState == LVIS_FOCUSED or LVIS_SELECTED) &&
([ebx].uOldState == 0)
                invoke EnableMenuItem, hMainMenu, IDM_SERV_EDIT,
MF_ENABLED
                invoke EnableMenuItem, hMainMenu,
IDM_SERV_DELETE, MF_ENABLED
                invoke EnableMenuItem, hMainMenu,
IDM_SERV_SECURITY, MF_ENABLED
                invoke EnableMenuItem, hMainMenu,
IDM_SERV_IOCONTROL, MF_ENABLED

                invoke SendMessage, hMainToolBar,
TB_ENABLEBUTTON, IDB_SERV_EDIT, TRUE
                invoke SendMessage, hMainToolBar,
TB_ENABLEBUTTON, IDB_SERV_DELETE, TRUE

                invoke UpdateServControls, hServiceList,
[ebx].iItem
            .endif
            assume ebx: ptr NMHDR
        .elseif ([ebx].code == NM_DBLCLK)
           invoke SendMessage, eax, LVM_GETNEXTITEM, -1,
LVNI_SELECTED or LVNI_FOCUSED
            .if (eax != -1)
                invoke EditServRecord, hServiceList
            .endif
        .elseif ([ebx].code == NM_SETFOCUS)
            push hServAccelerators
            pop hCurrentAccelerators
        .elseif ([ebx].code == NM_KILLFOCUS)
            push hMainAccelerators
            pop hCurrentAccelerators
        .endif
    .else
        assume ebx: ptr NMHDR
        .if ([ebx].code == TTN_GETDISPINFO)
            assume ebx: ptr TOOLTIPTEXT
```

```
            invoke LoadString, hInstance, [ebx].hdr.idFrom, addr
szTmpBuffer, TMP_BUF_SIZE
            lea eax, szTmpBuffer
            mov [ebx].lpszText, eax
        .endif
    .endif
    assume ebx: nothing
    pop ebx

    ret
OnNotify endp

;-------------------------------------------------------------
; UpdateServControls - оновити стан кнопок керування сервісом
; - hList – список сервісів
; - dwIndex - індекс запису
;-------------------------------------------------------------
UpdateServControls proc hList:HWND, dwIndex: DWORD
    LOCAL item:LV_ITEM
    LOCAL sStatus:SERVICE_STATUS

    LOCAL dwStartState: DWORD
    LOCAL dwStopState: DWORD
    LOCAL dwPauseState: DWORD
    LOCAL dwResumeState: DWORD
    LOCAL dwRestartState: DWORD

    push ebx

    xor eax, eax
    mov dwStartState, eax
    mov dwStopState, eax
    mov dwPauseState, eax
    mov dwResumeState, eax
    mov dwRestartState, eax

    mov eax, dwIndex
    .if (eax != -1)
        ; отримати ім'я сервісу
        mov item.iItem, eax
        mov item.imask, LVIF_TEXT
        mov item.iSubItem, 0
        lea eax, szTmpBuffer
        mov item.pszText, eax
        mov item.cchTextMax, TMP_BUF_SIZE
        invoke SendMessage, hList, LVM_GETITEM, 0, addr item

        .if (eax == TRUE)
            invoke GetServiceStatus, hSCManager, addr
szTmpBuffer, dwServAccess, addr sStatus
            lea ebx, sStatus
            assume ebx: ptr SERVICE_STATUS
            .if (eax == TRUE)
```

```
                mov eax, [ebx].dwControlsAccepted
                test eax, SERVICE_ACCEPT_PAUSE_CONTINUE
                .if (!ZERO?)
                    inc dwPauseState
                    inc dwResumeState
                .endif

                test eax, SERVICE_ACCEPT_STOP
                .if (!ZERO?)
                    inc dwStopState
                    inc dwRestartState
                .endif

                .if ([ebx].dwCurrentState == SERVICE_RUNNING)
                    xor eax, eax
                    mov dwStartState, eax
                    mov dwResumeState, eax
                    inc eax
                    and dwStopState, eax
                    and dwPauseState, eax
                    and dwRestartState, eax
                .elseif ([ebx].dwCurrentState ==
SERVICE_STOPPED)
                    xor eax, eax
                    mov dwStopState, eax
                    mov dwPauseState, eax
                    mov dwResumeState, eax
                    mov dwRestartState, eax
                    inc eax
                    mov dwStartState, eax
                .elseif ([ebx].dwCurrentState == SERVICE_PAUSED)
                    xor eax, eax
                    mov dwStartState, eax
                    mov dwPauseState, eax
                    inc eax
                    mov dwResumeState, eax
                    and dwStopState, eax
                    and dwRestartState, eax
                .endif
            .else
                xor eax, eax
                mov dwResumeState, eax
                inc eax
                mov dwStartState, eax
                mov dwPauseState, eax
                mov dwStopState, eax
                mov dwRestartState, eax
            .endif
        .endif
    .endif

    .if (dwStartState != FALSE)
        mov eax, MF_ENABLED
```

```
    .else
        mov eax, MF_GRAYED
    .endif
    invoke EnableMenuItem, hMainMenu, IDM_SERV_START, eax

    .if (dwStopState != FALSE)
        mov eax, MF_ENABLED
    .else
        mov eax, MF_GRAYED
    .endif
    invoke EnableMenuItem, hMainMenu, IDM_SERV_STOP, eax

    .if (dwPauseState != FALSE)
        mov eax, MF_ENABLED
    .else
        mov eax, MF_GRAYED
    .endif
    invoke EnableMenuItem, hMainMenu, IDM_SERV_PAUSE, eax

    .if (dwResumeState != FALSE)
        invoke SetToolBarButton, hMainToolBar, addr tbbuttons +
(BUTTONCOUNT+1)*sizeof TBBUTTON, dwStartButtonId
        mov dwStartButtonId, IDB_SERV_RESUME
        mov eax, MF_ENABLED
    .else
        invoke SetToolBarButton, hMainToolBar, addr tbbuttons +
6*sizeof TBBUTTON, dwStartButtonId
        mov dwStartButtonId, IDB_SERV_START
        mov eax, MF_GRAYED
    .endif
    invoke EnableMenuItem, hMainMenu, IDM_SERV_RESUME, eax

    .if (dwRestartState != FALSE)
        mov eax, MF_ENABLED
    .else
        mov eax, MF_GRAYED
    .endif
    invoke EnableMenuItem, hMainMenu, IDM_SERV_RESTART, eax

    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_START, dwStartState
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_STOP, dwStopState
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_PAUSE, dwPauseState
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESUME, dwResumeState
    invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESTART, dwRestartState

    pop ebx
    ret
UpdateServControls endp
```

```
;-------------------------------------------------------------
; CreateToolBar – функція створення головного тулбару
; - hParent - хендл батьківського вікна
; - hInstance - інстанс програми
; - pTBInfo - покажчик на структуру з інформацією про тулбар
;-------------------------------------------------------------
CreateToolBar proc hParent:HWND, hInst:HINSTANCE, pTBInfo:DWORD
    LOCAL hToolBar   :HWND
    LOCAL tbab         :TBADDBITMAP

    pushad

    mov ebx, pTBInfo
    mov eax, (ToolBarInfo ptr [ebx]).ItemWidth
    add eax, 10
    mul (ToolBarInfo ptr [ebx]).ItemCount
    mov esi, eax

    mov eax, (ToolBarInfo ptr [ebx]).ItemHeight
    add eax, 10

    invoke CreateWindowEx, 0, addr szToolBarClassName,
(ToolBarInfo ptr [ebx]).pWindowName, WS_CHILD OR WS_VISIBLE OR
TBSTYLE_FLAT OR TBSTYLE_TOOLTIPS ,\
                  0,0,esi, eax, hParent,0,hInst,0
    mov hToolBar, eax
    .if (eax)
        invoke SendMessage,
hToolBar,TB_BUTTONSTRUCTSIZE,sizeof(TBBUTTON),0
        mov  ecx,(ToolBarInfo ptr [ebx]).BitmapWidth
        mov  eax,(ToolBarInfo ptr [ebx]).BitmapHeight
        shl  eax,16
        mov  ax, cx
        invoke SendMessage, hToolBar,TB_SETBITMAPSIZE,0,eax;


        invoke LoadBitmap, hInst, (ToolBarInfo ptr
[ebx]).IdBitmap
        invoke SetBmpColor, eax
        mov (ToolBarInfo ptr [ebx]).hBitmap, eax

        mov tbab.hInst, 0
        mov tbab.nID, eax

        invoke SendMessage, hToolBar,TB_ADDBITMAP,(ToolBarInfo
ptr [ebx]).BitmapCount, addr tbab
        mov  ecx,(ToolBarInfo ptr [ebx]).ItemWidth
        mov  eax,(ToolBarInfo ptr [ebx]).ItemHeight
        shl  eax,16
        mov  ax, cx
```

```
        invoke SendMessage, hToolBar,TB_SETBUTTONSIZE,0,eax

        ; Створити кнопки
        invoke SendMessage, hToolBar,TB_ADDBUTTONS, BUTTONCOUNT,
(ToolBarInfo ptr [ebx]).pButtons
    .endif

    popad

    mov eax, hToolBar
    ret
CreateToolBar endp

;-------------------------------------------------------------
; SetBmpColor() - залити фон біт-мапу кольором кнопки
; - hBitmap - хендл бітмапа
;-------------------------------------------------------------
SetBmpColor proc  hBitmap:HBITMAP
    LOCAL mDC         :DWORD
    LOCAL hBrush      :DWORD
    LOCAL hOldBmp   :DWORD
    LOCAL hOldBrush :DWORD

    invoke CreateCompatibleDC,NULL
    mov mDC,eax

    invoke SelectObject,mDC,hBitmap
    mov hOldBmp,eax

    invoke GetSysColor,COLOR_BTNFACE
    invoke CreateSolidBrush,eax
    mov hBrush,eax

    invoke SelectObject,mDC,hBrush
    mov hOldBrush,eax

    invoke GetPixel,mDC,1,1
    invoke ExtFloodFill,mDC,1,1,eax,FLOODFILLSURFACE

    invoke SelectObject,mDC,hOldBrush
    invoke DeleteObject,hBrush

    invoke SelectObject,mDC,hBitmap
    push eax
    invoke DeleteDC,mDC
    pop eax

    ret
SetBmpColor endp

;-------------------------------------------------------------
; SetToolBarButton – замінити кнопку тулбару
; - hToolBar - хендл тулбара
```

```
; - pTButton – покажчик на структуру з кнопки
; - Id – індекс кнопки
;-------------------------------------------------------------
SetToolBarButton proc hToolBar: HWND, pTButton: DWORD, Id: DWORD
    LOCAL tbbi: TBBUTTONINFO
    LOCAL ButtonRect: RECT

    push ebx
    mov ebx, pTButton

    assume ebx: ptr TBBUTTON
    push sizeof TBBUTTONINFO
    pop tbbi.cbSize
    push TBIF_COMMAND OR TBIF_IMAGE
    pop tbbi.dwMask

    push [ebx].idCommand
    pop tbbi.idCommand
    push [ebx].iBitmap
    pop tbbi.iImage
    assume ebx: nothing

    invoke SendMessage, hToolBar,TB_GETITEMRECT, Id, addr
ButtonRect
    invoke SendMessage, hToolBar,TB_SETBUTTONINFO, Id, addr tbbi
    invoke InvalidateRect, hToolBar, addr ButtonRect, FALSE

    pop ebx
    ret
SetToolBarButton endp

;-------------------------------------------------------------
; CreateTreeView – функція створення дерева
; - hParent - хендл батьківського вікна
; - hInstance - інстанс програми
; - pTVInfo - покажчик на структуру з інформацією про тулбар
;-------------------------------------------------------------
CreateTreeView proc hParent:HWND, hInst:HINSTANCE, pTVInfo:DWORD
    LOCAL hTreeView   :HWND
    LOCAL tvi:  TV_INSERTSTRUCT
    LOCAL hBitmap: HBITMAP

    pushad

    ; создать  TreeView
    invoke CreateWindowEx, WS_EX_CLIENTEDGE, addr
szTreeViewClassName, 0, WS_CHILD or WS_VISIBLE or TVS_HASLINES
or TVS_HASBUTTONS or TVS_LINESATROOT or WS_CLIPSIBLINGS,\
                    ClientRect.left, ClientRect.top, uLeftWidth,
uTopHeight, hParent, 0, hInst, NULL
    mov  hTreeView, eax

    ; создать  ImageList
```

```
    mov ebx, pTVInfo
    assume ebx: ptr TreeViewInfo

    mov eax, [ebx].ImageCount
    shl eax, 1
    invoke ImageList_Create, [ebx].ImageWidth,
[ebx].ImageHeight, ILC_COLOR4 OR ILC_MASK, [ebx].ImageCount, eax
    mov [ebx].hImageList, eax
    invoke LoadBitmap, hInst, [ebx].IdBitmap
    mov hBitmap, eax
    invoke LoadBitmap, hInst, [ebx].IdMask
    push eax
    invoke ImageList_Add, [ebx].hImageList, hBitmap, eax
    invoke DeleteObject, hBitmap
    pop eax
    invoke DeleteObject, eax

    invoke SendMessage, hTreeView, TVM_SETIMAGELIST, 0,
[ebx].hImageList
    assume ebx: nothing

    ; створити елементи
    mov ebx, pTVInfo
    mov ebx, (TreeViewInfo ptr [ebx]).pItems
    assume ebx: ptr TreeViewItem

    invoke SendMessage, hTreeView, TVM_DELETEITEM, 0, TVI_ROOT
    push [ebx].hParent
    pop tvi.hParent
    mov tvi.hInsertAfter, TVI_ROOT
    mov tvi.item.imask, TVIF_TEXT or TVIF_IMAGE or
TVIF_SELECTEDIMAGE or TVIF_STATE or TVIF_PARAM
    mov tvi.item.state, TVIS_EXPANDED
    mov tvi.item.stateMask , TVIS_EXPANDED
    lea eax, [ebx].ItemString
    mov tvi.item.pszText, eax
    push [ebx].iImage
    pop tvi.item.iImage
    push [ebx].iSelectedImage
    pop tvi.item.iSelectedImage
    push [ebx].lParam
    pop tvi.item.lParam
    invoke SendMessage, hTreeView, TVM_INSERTITEM, 0, addr tvi
    mov [ebx].hItem, eax

    add ebx, sizeof TreeViewItem
    mov [ebx].hParent, eax
    push ebx
    add ebx, sizeof TreeViewItem
    mov [ebx].hParent, eax
    pop ebx

    mov tvi.hParent, eax
```

```
    mov tvi.hInsertAfter, TVI_LAST
    lea eax, [ebx].ItemString
    mov tvi.item.pszText, eax
    push [ebx].iImage
    pop tvi.item.iImage
    push [ebx].iSelectedImage
    pop tvi.item.iSelectedImage
    push [ebx].lParam
    pop tvi.item.lParam
    invoke SendMessage, hTreeView, TVM_INSERTITEM, 0, addr tvi
    mov [ebx].hItem, eax

    add ebx, sizeof TreeViewItem
    lea eax, [ebx].ItemString
    mov tvi.item.pszText, eax
    push [ebx].iImage
    pop tvi.item.iImage
    push [ebx].iSelectedImage
    pop tvi.item.iSelectedImage
    push [ebx].lParam
    pop tvi.item.lParam
    invoke SendMessage, hTreeView, TVM_INSERTITEM, 0, addr tvi
    mov [ebx].hItem, eax

    assume ebx: nothing
    popad

    mov eax, hTreeView
    ret
CreateTreeView endp

;-------------------------------------------------------------
; FreeResources - звільнення ресурсів
;-------------------------------------------------------------
FreeResources proc
    ; закриваємо з'єднання з SCM
    invoke UnLockSCM
    invoke CloseSCMConnection

    ; звільняємо пам'ять
    invoke mfree, pStatusInfo
    invoke mfree, pConfigInfo
    invoke mfree, pDescriptionInfo
    invoke mfree, pFailureInfo
    invoke mfree, pGroupInfo
    invoke mfree, cnCompData.lpCompName

    invoke DestroyAcceleratorTable, hMainAccelerators
    invoke DestroyAcceleratorTable, hServAccelerators

    ret
FreeResources endp
```

```
;-------------------------------------------------------------
; SCMConnectTo - з'єднання з віддаленим комп'ютером
;-------------------------------------------------------------
SCMConnectTo proc hWnd:HWND
    LOCAL item: TVITEM

    invoke DialogBoxParam, hInstance, IDD_CONNECT_TO, hWnd,
ConnectToDialogFunc, addr cnCompData
    .if (eax == IDC_OK)
        mov item._mask, TVIF_HANDLE or TVIF_TEXT
        push tvitems.hItem
        pop item.hItem
        .if (cnCompData.dwCompFlag == CND_LOCALCOMP)
            lea eax, tvitems.ItemString
        .else
            mov eax, cnCompData.lpCompName
        .endif
        mov item.pszText, eax
        invoke SendMessage, hMainTreeView, TVM_SETITEM, 0, addr
item

        .if (hSCManager != 0)
            invoke UnLockSCM
            invoke CloseSCMConnection
            invoke OpenSCMConnection
        .endif
    .endif
    ret
SCMConnectTo endp

;-------------------------------------------------------------
; ExportDataToFile - експортувати дані
; - hList - хендл контролем
;-------------------------------------------------------------
ExportDataToFile proc hList: HWND
    lea eax, szTmpBuffer
    mov byte ptr [eax], 0

    mov ofsExportPath.lStructSize, sizeof ofsExportPath
    push hList
    pop ofsExportPath.hwndOwner
    push hInstance
    pop ofsExportPath.hInstance
    mov ofsExportPath.lpstrFilter, offset  szExportFilter
    mov ofsExportPath.nFilterIndex, NULL
    mov ofsExportPath.lpstrFile, eax
    mov ofsExportPath.nMaxFile, TMP_BUF_SIZE
    mov ofsExportPath.lpstrDefExt, offset szDefTxtExt

    invoke GetSaveFileName, addr ofsExportPath
    .if (eax != 0)
        invoke ExportListToFile,  hList, addr szTmpBuffer
    .endif
```

```
    ret
ExportDataToFile endp

;-------------------------------------------------------------
; Exit – вийти з програми
;-------------------------------------------------------------
Exit proc hWnd:HWND
    .if (dwControlWindowCount == 0)
        Invoke DestroyWindow, hWnd
    .else
        invoke MessageBox, hWnd, addr szOnCloseMessage, addr
szAppName, MB_ICONWARNING or MB_OK
    .endif
    ret
Exit endp

;-------------------------------------------------------------
; OpenSCMAccess - відкрити діалог Потрібні права доступу до SCM
;-------------------------------------------------------------
OpenSCMAccess proc hWnd:HWND
    invoke DialogBoxParam, hInstance, IDD_SCM_ACCESS, hWnd,
SCMAccessDialogFunc, addr dwSCMAccess
    ret
OpenSCMAccess endp

;-------------------------------------------------------------
; OpenSCMConnection – відкрити з'єднання з SCM
;-------------------------------------------------------------
OpenSCMConnection proc
    .if (hSCManager == 0)
        xor eax, eax
        .if (cnCompData.dwCompFlag != CND_LOCALCOMP)
            mov eax, cnCompData.lpCompName
        .endif
        invoke OpenSCManager, eax, NULL, dwSCMAccess
        mov hSCManager, eax

        .if (eax != 0)
            invoke SetToolBarButton, hMainToolBar, addr
tbbuttons + BUTTONCOUNT*sizeof TBBUTTON, IDB_SCM_OPEN
        .endif

        invoke GetLastErrorString, addr szErrorDescription,
ERROR_DESC_SIZE, addr szErrorCode, ERROR_CODE_SIZE
        invoke LogMessage, hLogList, addr szSCM, addr
szOperationOpen, hSCManager, addr szErrorCode, addr
szErrorDescription

         ; обновить   список  сервисов
        invoke FillServicesList, hServiceList, hLogList,
hSCManager, dwServiceEnumType, SERVICE_STATE_ALL

        ; розблокувати органи управління
```

```
        invoke EnableMenuItem, hMainMenu, IDM_SCM_LOCK,
MF_ENABLED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_UNLOCK,
MF_ENABLED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_NEW,
MF_ENABLED
        invoke EnableMenuItem, hMainMenu, IDM_REFRESH,
MF_ENABLED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_OPEN,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_CLOSE,
MF_ENABLED

        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_NEW, TRUE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_REFRESH, TRUE
    .endif

    ret
OpenSCMConnection endp

;-------------------------------------------------------------
; CloseSCMConnection - закрити з'єднання з SCM
;-------------------------------------------------------------
CloseSCMConnection proc
    .if (hSCManager != 0)
        invoke CloseServiceHandle, hSCManager
        push eax
        .if (eax != 0)
            xor eax, eax
            mov hSCManager, eax

            invoke SetToolBarButton, hMainToolBar, addr
tbbuttons, IDB_SCM_CLOSE
        .endif

        invoke GetLastErrorString, addr szErrorDescription,
ERROR_DESC_SIZE, addr szErrorCode, ERROR_CODE_SIZE
        pop eax
        invoke LogMessage, hLogList, addr szSCM, addr
szOperationClose, eax, addr szErrorCode, addr szErrorDescription

         ; оновити список сервісів
        invoke FillServicesList, hServiceList, hLogList,
hSCManager, dwServiceEnumType, SERVICE_STATE_ALL

        ; разблокировать  органы управления
        invoke EnableMenuItem, hMainMenu, IDM_SCM_OPEN,
MF_ENABLED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_CLOSE,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_LOCK,
```

```
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SCM_UNLOCK,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_REFRESH,
MF_GRAYED

        invoke EnableMenuItem, hMainMenu, IDM_SERV_SECURITY,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_NEW,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_EDIT,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_DELETE,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_START,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_STOP,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_PAUSE,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_RESUME,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_RESTART,
MF_GRAYED
        invoke EnableMenuItem, hMainMenu, IDM_SERV_IOCONTROL,
MF_GRAYED

        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_NEW, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_EDIT, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_DELETE, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_START, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_STOP, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_PAUSE, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESUME, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_SERV_RESTART, FALSE
        invoke SendMessage, hMainToolBar, TB_ENABLEBUTTON,
IDB_REFRESH, FALSE
    .endif
    ret
CloseSCMConnection endp

;-------------------------------------------------------------
; LockSCM - заблокувати базу SCM
;-------------------------------------------------------------
LockSCM proc
```

```
    .if (hSCManager != 0) && (sclLock == 0)
        invoke LockServiceDatabase,hSCManager
        mov sclLock, eax

        invoke GetLastErrorString, addr szErrorDescription,
ERROR_DESC_SIZE, addr szErrorCode, ERROR_CODE_SIZE
        invoke LogMessage, hLogList, addr szSCM, addr
szOperationLock, sclLock, addr szErrorCode, addr
szErrorDescription
    .endif
    ret
LockSCM endp

;-------------------------------------------------------------
; UnLockSCM - розблокувати бвзу SCM
;-------------------------------------------------------------
UnLockSCM proc
    .if (sclLock != 0)
        invoke UnlockServiceDatabase,sclLock
        push eax
        .if (eax != 0)
            xor eax, eax
            mov sclLock, eax
        .endif

        invoke GetLastErrorString, addr szErrorDescription,
ERROR_DESC_SIZE, addr szErrorCode, ERROR_CODE_SIZE
        pop eax
        invoke LogMessage, hLogList, addr szSCM, addr
szOperationUnlock, eax, addr szErrorCode, addr
szErrorDescription
    .endif
    ret
UnLockSCM endp

;-------------------------------------------------------------
; OpenServAccess - відкрити дилалог Необхідні права доступу
;-------------------------------------------------------------
OpenServAccess proc hWnd:HWND
    invoke DialogBoxParam, hInstance, IDD_SERV_ACCESS, hWnd,
ServAccessDialogFunc, addr dwServAccess
    ret
OpenServAccess endp

;-------------------------------------------------------------
; OpenServSecurity - відкрити діалог Безпека
;-------------------------------------------------------------
OpenServSecurity proc hWnd:HWND
    ret
OpenServSecurity endp

;-------------------------------------------------------------
; AddServRecord - надати новий сервіс на базу SCM
```

```
; - hList - хендл списку сервісів
;-------------------------------------------------------------
AddServRecord proc hList: HWND
    .if (hSCManager != 0)
        invoke malloc, sizeof ServiceDataParam
        .if (eax != 0)
            mov ebx, eax
            assume ebx: ptr ServiceDataParam

            push hSCManager
            pop [ebx].hSCManager
            push dwServiceEnumType
            pop [ebx].dwServiceType
            push dwServAccess
            pop [ebx].dwDesiredAccess
            mov [ebx].dwMode, DIALOGPARAM_NEW
            push hLogList
            pop [ebx].hLog
            push cnCompData.dwCompFlag
            pop [ebx].cdComp.dwCompFlag
            push cnCompData.dwCompNameSize
            pop [ebx].cdComp.dwCompNameSize
            push cnCompData.lpCompName
            pop [ebx].cdComp.lpCompName
            lea eax, dwControlWindowCount
            mov [ebx].lpdwCount, eax

            invoke DialogBoxParam, hInstance,
IDD_SERVICE_PROPERTIES, hList, ServicePropertiesDialogFunc, ebx
            .if (eax != IDC_CANCEL)
                invoke RefreshData, hList
            .endif
            invoke SetFocus, hList
        .endif
        assume ebx: nothing
        pop ebx
    .endif
    ret
AddServRecord endp

;-------------------------------------------------------------
; EditServRecord - редагування властивостей сервісу
; - hList - хендл списку сервісів
;-------------------------------------------------------------
EditServRecord proc hList: HWND
    LOCAL dwIndex: DWORD
    LOCAL item:LV_ITEM

    invoke SendMessage, hList,LVM_GETNEXTITEM, -1, LVNI_SELECTED
or LVNI_FOCUSED
    .if (hSCManager != 0) && (eax != -1)
        mov dwIndex, eax
        push ebx
```

```
        invoke malloc, sizeof ServiceDataParam
        .if (eax != 0)
            mov ebx, eax
            assume ebx: ptr ServiceDataParam

            push hSCManager
            pop [ebx].hSCManager
            push dwServiceEnumType
            pop [ebx].dwServiceType
            push dwServAccess
            pop [ebx].dwDesiredAccess
            mov [ebx].dwMode, DIALOGPARAM_EDIT
            push hLogList
            pop [ebx].hLog
            push cnCompData.dwCompFlag
            pop [ebx].cdComp.dwCompFlag
            push cnCompData.dwCompNameSize
            pop [ebx].cdComp.dwCompNameSize
            push cnCompData.lpCompName
            pop [ebx].cdComp.lpCompName
            lea eax, dwControlWindowCount
            mov [ebx].lpdwCount, eax

            ; получить  имя сервиса
            push dwIndex
            pop item.iItem
            mov item.imask, LVIF_TEXT
            mov item.iSubItem, 0
            lea eax, [ebx].szServiceName
            mov item.pszText, eax
            mov item.cchTextMax, SERV_NAME_LEN
            invoke SendMessage, hList, LVM_GETITEM, 0, addr item

            .if (eax == TRUE)
                invoke DialogBoxParam, hInstance,
IDD_SERVICE_PROPERTIES, hList, ServicePropertiesDialogFunc, ebx
                .if (eax != IDC_CANCEL)
                    invoke UpdateServiceRecord, hList, dwIndex,
hSCManager
                    invoke UpdateServControls, hList, dwIndex
                .endif
                invoke SetFocus, hList
            .else
                invoke mfree, ebx
            .endif
        .endif
        assume ebx: nothing
        pop ebx
    .endif
    ret
EditServRecord endp

;-------------------------------------------------------------
```

```
; DeleteServRecord - видалити сервіс із бизу SCM
; - hList - хендл списку сервісів
;-------------------------------------------------------------
DeleteServRecord proc hList: HWND
    LOCAL dwIndex: DWORD
    LOCAL item:LV_ITEM

    invoke SendMessage, hList,LVM_GETNEXTITEM, -1, LVNI_SELECTED
or LVNI_FOCUSED
    .if (hSCManager != 0) && (eax != -1)
        ; отримати ім'я сервісу
        mov dwIndex, eax
        mov item.iItem, eax
        mov item.imask, LVIF_TEXT
        mov item.iSubItem, 0
        lea eax, szTmpBuffer
        mov item.pszText, eax
        mov item.cchTextMax, TMP_BUF_SIZE
        invoke SendMessage, hList, LVM_GETITEM, 0, addr item

        .if (eax == TRUE)
            invoke wsprintf, addr szTmpBuffer2, addr
szDeleteServiceMessage, addr szTmpBuffer
            invoke MessageBox, hList, addr szTmpBuffer2, addr
szAppName, MB_ICONQUESTION or MB_YESNO
            .if(eax == IDYES)
                invoke ServiceDeleteOperation, hLogList,
hSCManager, addr szTmpBuffer, dwServAccess
                .if (eax != 0)
                    invoke RefreshData, hList
                .endif
            .endif
            invoke SetFocus, hList
        .endif
    .endif

    ret
DeleteServRecord endp

;-------------------------------------------------------------
; ServControl - керування сервісом
; - hList - хендл списку сервісів
; - lpControlFunction
;-------------------------------------------------------------
ServControl proc hList: HWND, lpControlFunction: DWORD
    LOCAL dwIndex: DWORD
    LOCAL item:LV_ITEM

    invoke SendMessage, hList,LVM_GETNEXTITEM, -1, LVNI_SELECTED
or LVNI_FOCUSED
    .if (eax != -1)
        ; получить  имя сервиса
        mov dwIndex, eax
```

```
        mov item.iItem, eax
        mov item.imask, LVIF_TEXT
        mov item.iSubItem, 0
        lea eax, szTmpBuffer
        mov item.pszText, eax
        mov item.cchTextMax, TMP_BUF_SIZE
        invoke SendMessage, hList, LVM_GETITEM, 0, addr item

        .if (eax == TRUE)
            push dwServAccess
            lea eax, szTmpBuffer
            push eax
            push hSCManager
            push hLogList
            call lpControlFunction
            invoke UpdateServiceRecord, hList, dwIndex,
hSCManager
            invoke UpdateServControls, hList, dwIndex
        .endif
    .endif
    ret
ServControl endp

;-------------------------------------------------------------
; ServIOControl - надіслати сервісу керуючий код
; - hWnd - хендл вікна
; - hList - хендл списку сервісів
;-------------------------------------------------------------
ServControlDlg proc hWnd:HWND, hList: HWND
    LOCAL dwIndex: DWORD
    LOCAL item:LV_ITEM

    invoke SendMessage, hList,LVM_GETNEXTITEM, -1, LVNI_SELECTED
or LVNI_FOCUSED
    .if (hSCManager != 0) && (eax != -1)
        mov dwIndex, eax
        push ebx
        invoke malloc, sizeof ServiceDataParam
        .if (eax != 0)
            mov ebx, eax
            assume ebx: ptr ServiceDataParam

            push hSCManager
            pop [ebx].hSCManager
            push dwServiceEnumType
            pop [ebx].dwServiceType
            push dwServAccess
            pop [ebx].dwDesiredAccess
            mov [ebx].dwMode, DIALOGPARAM_EDIT
            push hLogList
            pop [ebx].hLog
            push cnCompData.dwCompFlag
            pop [ebx].cdComp.dwCompFlag
```

```
            push cnCompData.dwCompNameSize
            pop [ebx].cdComp.dwCompNameSize
            push cnCompData.lpCompName
            pop [ebx].cdComp.lpCompName
            lea eax, dwControlWindowCount
            mov [ebx].lpdwCount, eax

            ; отримати ім'я сервісу
            push dwIndex
            pop item.iItem
            mov item.imask, LVIF_TEXT
            mov item.iSubItem, 0
            lea eax, [ebx].szServiceName
            mov item.pszText, eax
            mov item.cchTextMax, SERV_NAME_LEN
            invoke SendMessage, hList, LVM_GETITEM, 0, addr item

            .if (eax == TRUE)
                invoke DialogBoxParam, hInstance,
IDD_SERV_CONTROL, hWnd, ServiceControlDialogFunc, ebx
                ;invoke CreateDialogParam, hInstance,
IDD_SERV_CONTROL, hWnd, ServiceControlDialogFunc, ebx
                ;invoke ShowWindow, eax, SW_SHOWNORMAL
            .else
                invoke mfree, ebx
            .endif
        .endif
        assume ebx: nothing
        pop ebx
    .endif
    ret
ServControlDlg endp

;-------------------------------------------------------------
; RefreshData - оновити дані
; - hList - хендл списку сервісів
;-------------------------------------------------------------
RefreshData proc hList: HWND
    LOCAL dwIndex: DWORD
    LOCAL dwTopIndex: DWORD
    LOCAL dwHeight: DWORD
    LOCAL rItemRect: RECT
    LOCAL item:LV_ITEM

    mov item.imask, LVIF_STATE
    mov item.stateMask, LVNI_SELECTED or LVNI_FOCUSED
    mov item.state, LVNI_SELECTED or LVNI_FOCUSED

    invoke SendMessage, hList,LVM_GETNEXTITEM, -1, LVNI_SELECTED
or LVNI_FOCUSED
    mov dwIndex, eax

    invoke SendMessage, hList,LVM_GETTOPINDEX, 0, 0
```

```
    mov dwTopIndex, eax

    mov rItemRect.left, LVIR_BOUNDS
    invoke SendMessage, hList, LVM_GETITEMRECT, dwTopIndex, addr
rItemRect
    mov eax, rItemRect.bottom
    sub eax, rItemRect.top
    mov dwHeight, eax

    invoke FillServicesList, hServiceList, hLogList, hSCManager,
dwServiceEnumType, SERVICE_STATE_ALL

    mov eax, dwTopIndex
    mul dwHeight
    invoke SendMessage, hList, LVM_SCROLL, 0, eax

    .if (dwIndex != -1)
        invoke SendMessage, hList, LVM_SETITEMSTATE, dwIndex,
addr item
    .endif

    ret
RefreshData endp

;-------------------------------------------------------------
; OpenAboutDlg - про програму
;-------------------------------------------------------------
OpenAboutDlg proc hWnd:HWND
    invoke DialogBoxParam, hInstance, IDD_ABOUT, hWnd,
AboutDialogFunc, NULL
    ret
OpenAboutDlg endp

;-------------------------------------------------------------
; ExportListToFile - експортувати дані з листям контролю в файл
; - hList - хендл контролем
; - lpFileName - ім'я файлу
;-------------------------------------------------------------
ExportListToFile proc hList: HWND, lpFileName: LPCSTR
    LOCAL hFile: HANDLE
    LOCAL dwColumnCount: DWORD
    LOCAL dwItemCount: DWORD
    LOCAL lpMem:LPVOID

    pushad
    invoke CreateFile, lpFileName, GENERIC_READ or
GENERIC_WRITE, FILE_SHARE_READ,
NULL,CREATE_ALWAYS,FILE_ATTRIBUTE_ARCHIVE, NULL
    mov hFile, eax
    .if (eax != INVALID_HANDLE_VALUE)
        invoke malloc, 2048
        .if (eax != 0)
            mov lpMem, eax
```

```
            invoke mfree, lpMem
        .endif
        invoke CloseHandle, hFile
    .endif

    popad
    ret
ExportListToFile endp

;-------------------------------------------------------------
; ServiceListProc - віконна процедура listview списку сервісів
; - hWnd - хендл вікна
; - uMsg - повідомлення
; - wParam – перший параметр повідомлення
; - lParam – другий параметр повідомлення
;-------------------------------------------------------------
ServiceListProc proc hWnd:HWND, uMsg:UINT, wParam:WPARAM,
lParam:LPARAM
    LOCAL rRect: RECT

    .if (uMsg == WM_CONTEXTMENU)
        mov eax, lParam
        mov ecx, eax
        and eax, 0FFFFh
        shr ecx, 16
        .if (ax == -1) && (cx == -1)
            invoke GetWindowRect, hWnd, addr rRect
            mov eax, rRect.left
            add eax, 20
            mov ecx, rRect.top
            add ecx, 20
        .endif
        invoke TrackPopupMenu, hServPopupMenu, TPM_LEFTALIGN,
eax, ecx, NULL, hWnd, NULL
    .elseif (uMsg == WM_COMMAND)
        mov eax, wParam
        and eax, 0ffffh
        .if (eax == IDM_SERV_ACCESS)
            invoke OpenServAccess, hWnd
        .elseif (eax == IDM_SERV_SECURITY)
            invoke OpenServSecurity, hWnd
        .elseif (eax == IDM_SERV_NEW)
            invoke AddServRecord, hWnd
        .elseif (eax == IDM_SERV_EDIT)
            invoke EditServRecord, hWnd
        .elseif (eax == IDM_SERV_DELETE)
            invoke DeleteServRecord, hWnd
        .elseif (eax == IDM_SERV_START)
            invoke ServControl, hWnd, ServiceStartOperation
        .elseif (eax == IDM_SERV_STOP)
            invoke ServControl, hWnd, ServiceStopOperation
        .elseif (eax == IDM_SERV_PAUSE)
```

```
            invoke ServControl, hWnd, ServicePauseOperation
        .elseif (eax == IDM_SERV_RESUME)
            invoke ServControl, hWnd, ServiceResumeOperation
        .elseif (eax == IDM_SERV_RESTART)
            invoke ServControl, hWnd, ServiceRestartOperation
        .elseif (eax == IDM_SERV_IOCONTROL)
            invoke ServControlDlg, hMainWnd, hWnd
        .elseif (eax == IDM_REFRESH)
            invoke RefreshData, hWnd
        .endif
    .endif

    invoke GetWindowLong, hWnd, GWL_USERDATA
    invoke CallWindowProc, eax, hWnd, uMsg, wParam, lParam
    ret
ServiceListProc endp

end begin
```

Б.2 Лістинг файлу опису та функцій main.inc до проекту

```
;-------------------------------------------------------------
; Керування драйверами Windows
;
; Файл: main.inc
; Опис: Опис типів і функцій, що використовуються в роботі з
вікном програми
; Автор: Шебанов Євгеній Олександрович
;-------------------------------------------------------------

; константи
MIN_WIDTH       EQU 320
MIN_HEIGHT      EQU 200

BUTTONCOUNT     EQU 14
TVI_COUNT       EQU 3
TB_SIZE         EQU 16
TVI_SIZE        EQU 16
TREE_WIDTH      EQU 120
LOG_HEIGHT      EQU 100
SPLITTER_SIZE   EQU 4
MIN_SIZE        EQU 20

SEVR_COLUMN_COUNT  EQU 9
LOG_COLUMN_COUNT   EQU 5

;-------------------------------------------------------------
; Функції
;-------------------------------------------------------------
WinMain proto hInst:HINSTANCE, hPrevInst:HINSTANCE,
lpCmdLine:LPSTR, nCmdShow:DWORD
MainWndProc proto hWnd:HWND, uMsg:UINT, wParam:WPARAM,
lParam:LPARAM
```

```
OnCreate proto hWnd:HWND
OnSize proto hWnd:HWND
GetMinMaxInfo proto lParam:LPARAM
OnLButtonDown proto hWnd:HWND, wParam:WPARAM, lParam:LPARAM
OnLButtonUp proto hWnd:HWND, wParam:WPARAM, lParam:LPARAM
OnMouseMove proto hWnd:HWND, wParam:WPARAM, lParam:LPARAM
OnNotify proto hWnd:HWND, wParam:WPARAM, lParam:LPARAM
UpdateServControls proto hList:HWND, dwIndex: DWORD

CreateToolBar proto hParent:HWND, hInstance:HINSTANCE,
pTBInfo:DWORD
SetBmpColor proto  hBitmap:HBITMAP
SetToolBarButton proto hToolBar: HWND, pTButton: DWORD, Id:
DWORD
CreateTreeView proto hParent: HWND, hInstance: HINSTANCE,
pTVInfo: DWORD

FreeResources proto

SCMConnectTo proto hWnd:HWND
ExportDataToFile proto hList: HWND
Exit proto hWnd:HWND

OpenSCMAccess proto hWnd:HWND

OpenSCMConnection proto
CloseSCMConnection proto
LockSCM proto
UnLockSCM proto

OpenServAccess proto hWnd:HWND
OpenServSecurity proto hWnd:HWND

AddServRecord proto hList: HWND
EditServRecord proto hList: HWND
DeleteServRecord proto hList: HWND

ServControl proto hList: HWND, lpControlFunction: DWORD
ServControlDlg proto hWnd:HWND, hList: HWND
RefreshData proto hList: HWND

OpenHelp proto hWnd:HWND
OpenAboutDlg proto hWnd:HWND

ExportListToFile proto hList: HWND, lpFileName: LPCSTR

ServiceListProc proto hWnd:HWND, uMsg:UINT, wParam:WPARAM,
lParam:LPARAM
```