

Operációs rendszerek BSC

3B. gyak.

2021. 02. 24.

Készítette:

Miliczki József Bsc

GÉIK - Programtervező Informatikus

Y86I0I

Miskolc, 2021

4. Feladat

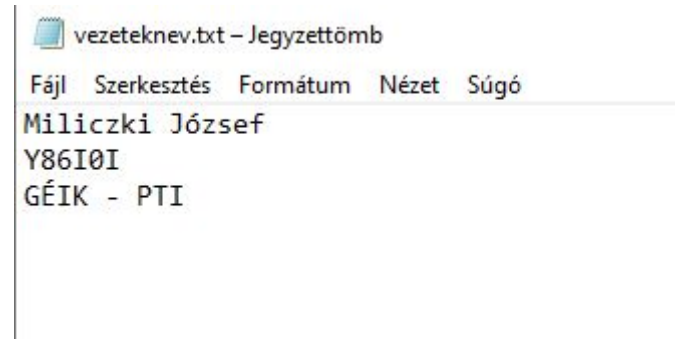
1) Készítsen egy neptunkod.c nevű forráskódot, amely egy vezeteknev.txt fájlt létrehoz, olvas, majd bezár. Tartalma: Név, Szak, Neptunkod etc.

Írtam egy rendkívül egyszerű kódot amely teljesíti a fenti kritériumokat:

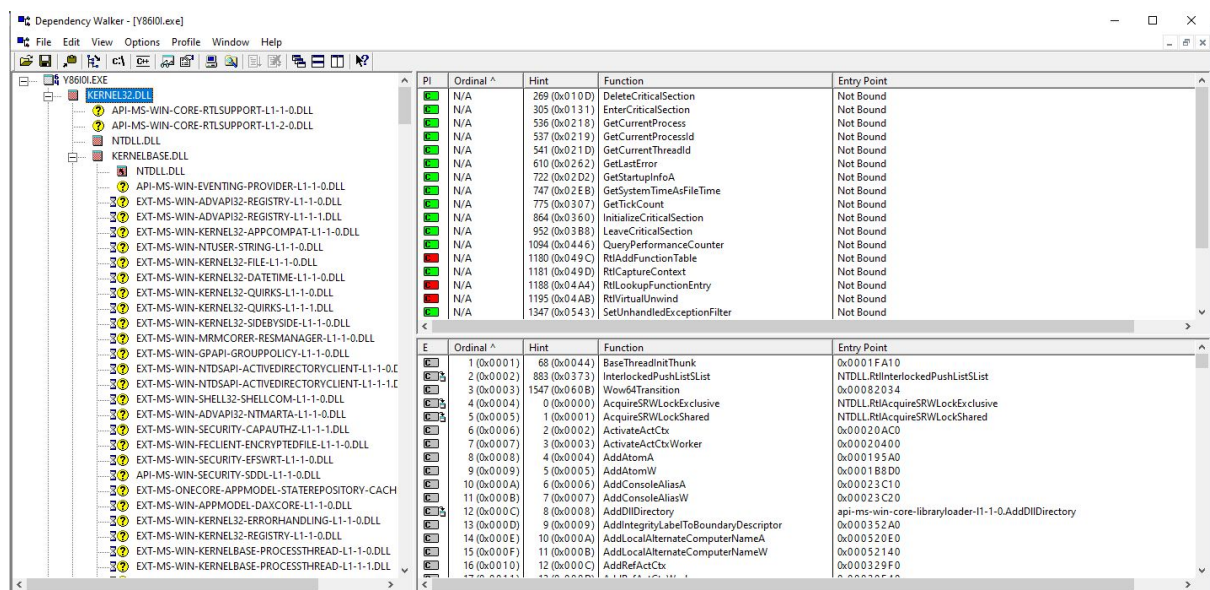
```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    FILE * file;
    char s[300];
    file = fopen("vezeteknev.txt", "r+");
    for (int i=0; i<4; i++)
    {
        fscanf(file, "%s", s);
        printf("%s\n", s);
    }
    return 0;
}
```

A vezeteknev.txt tartalma:



Nyissuk meg Dependency Walker-el a most legyártott programunkat:



a.) Vizsgálja meg, hogy a neptunkod.exe milyen API hívásokat használ a kernel32.dll-ből (Win alrendszer DLL)!

A fenti képből is látható hogy ténylegesen mennyi DLL-t használ a kernel32, ezekből jó párat ki fogok most emelni, és megvizsgálni milyen szolgáltatásokhoz engednek hozzá:

- **API-MS-WIN-CORE-PROCESSTHREADS-L1-1-0.DLL** - Ez az OS magja, és alapvető DLL file. Ezt abból lehet tudni hogy a Windows 7-es architektúra óta API-MS-WIN-el kezdődik minden olyan DLL file ami gyárilag jön az OS-el. Természetesen ez azt jelenti hogy vannak kívülről hozott Dynamic Link Library-k is.
- **API-MS-WIN-CORE-REGISTRY-L1-1-0.DLL** - A Windows registry kezelője. Számos funkcióhívás történik a System32-vel ezért a file hiánya súlyos következményekkel járhat. Maga a DLL 2012-ben készült Windows 8-ra, és utolsó frissítése 2019-ben jött ki az Oracle féle virtual machine-ra.
- **API-MS-WIN-CORE-HEAP-L1-1-0.DLL** - A Windows veremkezelő DLL-e. Olyan funkciókat találunk benne mint az Alloc és a Free. Azért ezeket emelem ki mert a fogalom előjön C programozáskor is, és funkciójuk identikus. A Windows 8-al jött be

- **API-MS-WIN-CORE-CONSOLE-L1-1-0.DLL** - A konzol működtetéséért felelős DLL. A hivatalos Microsoft dokumentáció szerint olyan funkciókat tartalmaz, amely a megjeleníthető karakterekkel foglalkozik, valamint a bevitt karaktereket fordítja le egy megadott kódtábla alapján. Nem meglepő, de főleg a bevitel és kiírás a feladata ennek a DLL-nek.
- **API-MS-WIN-CORE-ERRORHANDLING-L1-1-0.DLL** - Egy másik nagyon fontos DLL; hibakezelés. Olyan funkciókat találhatunk itt, amelyek azt biztosítják, hogy a processzek le tudják kommunikálni a problémákat, valamint végső esetben elindítsanak egy folyamatot amely eljuttatja az Error kódot a felhasználó irányába. Több módja van a hiba lekommunikálásának, a GetLastError például 4 féle hibát jelezhet.

Rengeteg más DDL található még a listán, viszont nem sorolhatom fel mindet. Lényegében alapvető Windows DLL file-okat használ a programom, amely természetesen nem meglepő.

b.) Milyen függőségei vannak a kernel32.dll-nek!

Őszinte leszek, ezt nagyon sokáig kutattam otthon és nem találtam rá kielégítő választ. A kernel32.dll-nek lehetséges hogy nincsenek függőségei? Inkább tőle függenek DLL-ek, és nem fordítva? Ha a kérdés a függvényekre gondol, akkor tudom mit kell keresnem, de a Microsoft oldala se adott útmutatást....

c.) Keresse meg NTDLL.DLL-t! Mi ennek a szerepe? Vizsgálja meg az exportált függvényeket, milyen információkat kap az NT API-ról!

Dependency Walker - [Y860LX.exe]					
File Edit View Options Profile Window Help					
Y860LX.EXE					
KERNEL32.DLL	N/A	Ordinal	Hint	Function	Entry Point
APRMS-WIN-CORE-RTL-SUPPORT-L1-1-0-DLL	N/A	34 (0x0022)		DbgPrint	Not Bound
APRMS-WIN-CORE-RTL-SUPPORT-L1-1-0-DLL	N/A	233 (0x0A5D)		Audit	Not Bound
APRMS-WIN-CORE-RTL-SUPPORT-L1-1-0-DLL	N/A	1510 (0x5F6)		PtUnhandlExcepionFilter	Not Bound
NTUSERUI.dll	N/A	630 (0x276)		NtTerminateProcess	Not Bound
APRMS-WIN-CORE-RTL-SUPPORT-L1-1-0-DLL	N/A	2421 (0x975)		wcscopy	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	2422 (0x976)		wcsncpy	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	114 (0x0072)		GetLocalResourceID	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	1338 (0x544)		RtlReadThreadProfilingData	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	1338 (0x545)		RtlQueryThreadProfiling	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	916 (0x394)		RtlDisableThreadProfiling	Not Bound
APRMS-WIN-CORE-PROCESS-THREADS-L1-1-0-DLL	N/A	1378 (0x54F)		RtlSetStatusToEnumNoTab	Not Bound
APRMS-WIN-CORE-REGISTRY-L1-1-0-DLL	N/A	938 (0x3AA)		RtlEnableThreadProfiling	Not Bound
APRMS-WIN-CORE-HEAP-L1-1-0-DLL	N/A	401 (0x195)		RtlMapUserPhysicalPagesScatter	Not Bound
APRMS-WIN-CORE-HEAP-L1-1-0-DLL	N/A	800 (0x310)		RtlDecommitMemory	Not Bound
APRMS-WIN-CORE-MEMORY-L1-1-0-DLL	N/A	2331 (0x91B)		Search	Not Bound
APRMS-WIN-CORE-MEMORY-L1-1-0-DLL	N/A	889 (0x359)		RtlComputeSystemTableHash	Not Bound
APRMS-WIN-CORE-MEMORY-L1-1-0-DLL	N/A	985 (0x3D9)		RtlEndActivationContextSectionId	Not Bound
APRMS-WIN-CORE-HANDLE-L1-1-0-DLL	N/A	1301 (0x515)		RtlQueryActivationContextSectionSettings	Not Bound
APRMS-WIN-CORE-SYNC-L1-1-0-DLL	N/A	1301 (0x516)		RtlIsSubAuthorityCount	Not Bound
APRMS-WIN-CORE-SYNCH-L1-2-0-DLL	N/A	135 (0x008)		GetLastErrorFromDirectory	Not Bound
APRMS-WIN-CORE-SYNCH-L1-2-0-DLL	N/A	147 (0x009)		RtlSecureHeap	Not Bound
APRMS-WIN-CORE-FILE-L1-1-0-DLL	N/A	1386 (0x55A)		RtlGetCurrentCultureNameAsTlsCb	Not Bound
APRMS-WIN-CORE-FILE-L1-2-0-DLL	N/A	487 (0x1E7)		NtQueryInstallLanguage	Not Bound
APRMS-WIN-CORE-FILE-L1-2-0-DLL	N/A	971 (0x3C8)		RtlExpandEnvironmentStringsU	Not Bound
APRMS-WIN-CORE-FILE-L1-2-0-DLL	E	Ordinal #	Hint	Function	Entry Point
APRMS-WIN-CORE-DELEGATE-L1-1-0-DLL	8	(0x0003)	916 (0x394)	RtlDispatchApc	0x0002C0A0
APRMS-WIN-CORE-IO-L1-1-0-DLL	9	(0x0009)	711 (0x2C7)	RtlActivateActivationContextUnsafeFast	0x0003F490
APRMS-WIN-CORE-IO-L1-1-0-DLL	10	(0x000A)	876 (0x36C)	RtlDeactivateActivationContextUnsafeFast	0x00043170
APRMS-WIN-CORE-Kernel-L1-1-0-DLL	11	(0x000B)	1186 (0x4AE)	PrintUnicodeStringList	0x00083800
APRMS-WIN-CORE-THREADPOOL-LEGACY-L1-1-0-DLL	12	(0x000C)	1500 (0x5E4)	RtlWorkByQueueWap	0x0008EB40
APRMS-WIN-CORE-THREADPOOL-PRIVATE-L1-1-0-DLL	13	(0x000D)	1509 (0x5E3)	RtlWorkByQueueWap	0x0008EB80
APRMS-WIN-CORE-THREADPOOL-PRIVATE-L1-1-0-DLL	14	(0x000E)	1551 (0x611)	RtlWorkByQueueWap	0x0008EB80
APRMS-WIN-CORE-LIBRARY-LOADER-L1-2-0-DLL	15	(0x000F)	0 (0x0000)	A_SHMInit	0x00067740
APRMS-WIN-CORE-LIBRARY-LOADER-L1-2-0-DLL	16	(0x0010)	1 (0x0001)	A_SHMinit	0x00087550
APRMS-WIN-CORE-LIBRARY-LOADER-L1-2-0-DLL	17	(0x0011)	2 (0x0002)	A_SHMinit	0x00067520
APRMS-WIN-CORE-LIBRARY-LOADER-L1-2-0-DLL	18	(0x0012)	3 (0x0003)	AlgeAcJustCompletionListConcurrencyCount	0x0008BE60
APRMS-WIN-CORE-NAMEDPIPE-L1-2-0-DLL	19	(0x0013)	4 (0x0004)	AlgeFreeCompletionListMessage	0x0008B110
APRMS-WIN-CORE-NAMEDPIPE-L1-2-0-DLL	20	(0x0014)	5 (0x0005)	AlgeGetCompletionListAndMessageInformation	0x0008BA00
APRMS-WIN-CORE-NAMEDPIPE-L1-2-0-DLL	21	(0x0015)	6 (0x0006)	AlgeGetCompletionListMessageAttributes	0x0008BA30
APRMS-WIN-CORE-DATETIME-L1-1-0-DLL	22	(0x0016)	7 (0x0007)	AlgeGetIdateSize	0x00069170
APRMS-WIN-CORE-DATETIME-L1-1-0-DLL	23	(0x0017)	8 (0x0008)	AlgeGetMessageAttribute	0x00069180
APRMS-WIN-CORE-DATETIME-L1-1-0-DLL	24	(0x0018)	9 (0x0009)	AlgeGetMessageFromCompletionList	0x0006A070
APRMS-WIN-CORE-DATETIME-L1-1-0-DLL	25	(0x0019)	10 (0x000A)	AlgeGetCompletionListMessageCount	0x0006A080
APRMS-WIN-CORE-SYSTEM-L1-2-0-DLL	26	(0x001A)	11 (0x000B)	AlgeInitialzeMessageAttribute	0x000690F0
APRMS-WIN-CORE-SYSTEM-L1-2-0-DLL	27	(0x001B)	12 (0x000C)	AlgeMaxAllowedMessageLength	0x0006BE60
APRMS-WIN-CORE-SYSTEM-L1-2-0-DLL	28	(0x001C)	13 (0x000D)	AlgeRegisterCompletionList	0x0006BF80
APRMS-WIN-CORE-SYSTEM-L1-2-0-DLL	29	(0x001D)	14 (0x000E)	AlgeRegisterCompletionListWorkerThread	0x0006BC40
APRMS-WIN-CORE-SYSTEM-L1-2-0-DLL	30	(0x001E)	15 (0x000F)	AlgeRandomCompletionList	0x0006CB80
APRMS-WIN-CORE-TIMEZONE-L1-1-0-DLL	31	(0x001F)	1	AlgeRegisterCompletionList	0x0006CB80

Az **NTDLL.DLL** egy olyan erőforrás, amely nem a Win32 Subsystem része, pontosabban miatta létezhetnek az “egy réteggel” feljebb elhelyezkedő DLL fájloknak. Mivel technikailag nem alapszik egyik erőforrásfájlon sem, Natív API-nak is szokták nevezni. Használata főleg akkor kerül elő, mikor a rendszer elindul, és még nem állt fel a Win32 Subsystem. Mivel funkcionalitását megosztja a Kernel32-vel, így a rendszer felállása után “nagyon kevés” applikáció használja magát az NTDLL.DLL-t. Fundamentális természete miatt az NTDLL futtatható fájlok nem indíthatóak el a felhasználó által. Az ilyen fájlokat natív fájloknak nevezzük, ilyen például az **autochk.exe**, amely a jól ismert “kék halál”-nál indul el. Legkönnyebb az NTDLL.DLL-re úgy gondolni, mint egy lépcsőre ami utat nyit a ténylegesen runtime idő alatt használt DLL-ek számára!