**Name: Mukul Rai**

**Student ID: 700748568**

**GitHub Link for Project:** [**https://github.com/raimukul/Malware\_Project**](https://github.com/raimukul/Malware_Project)

**Project 1**

You should parse the attached two applications for their DOS headers and NT headers (including signature, coff header, and optional header) and print out their values in a format like (fields name: field value). The codes should be written in C or C++. You can also download the applications at <https://github.com/squarekyzhong/project1-3.git>

For example, Dos Header has 19 fields; you print like

e\_magic: 5A4D

e\_cblp: 90

e\_cp: 3

e\_crlc：0

e\_cparhdr: 4

…

Similarly, for NT headers.

All values should be in hexadecimal.

You should also download the Firefox browser to download the applications because Google Chrome will check them as malicious. <https://www.mozilla.org/en-US/firefox/new/>

**Code (Using C Programming Language)**

#include <stdio.h>

#include <windows.h>

#define **REDFONT** "\x1B[31m"

#define **GRNFONT** "\x1B[32m"

*int* **main**(*int* *argc*, *char* \**argv*[])

{

    if (*argc* != 2)

    {

**printf**("Usage: %s <filename> \n \n", *argv*[0]);

        return 1;

    }

***FILE*** \*file = **fopen**(*argv*[1], "rb");

    if (file == **NULL**)

    {

**printf**("Error: Unable to open application you provided '%s'\n", *argv*[1]);

        return 1;

    }

    // This below code will read the DOS header of an application

***IMAGE\_DOS\_HEADER*** dos\_header;

**fread**(&dos\_header, sizeof(dos\_header), 1, file);

    // This below code will verify the DOS signature of an application

    if (dos\_header.e\_magic != **IMAGE\_DOS\_SIGNATURE**)

    {

**printf**("Error: Invalid DOS signature of application \n");

**fclose**(file);

        return 1;

    }

    // This below code will print out the DOS header fields of your application

**printf**("\n" **REDFONT**);

**printf**("DOS header details below: \n" **GRNFONT**);

**printf**("e\_magic: % 4X\n", dos\_header.e\_magic);

**printf**("e\_cblp: % 4X\n", dos\_header.e\_cblp);

**printf**("e\_cp: % 4X\n", dos\_header.e\_cp);

**printf**("e\_crlc: % 4X\n", dos\_header.e\_crlc);

**printf**("e\_cparhdr: % 4X\n", dos\_header.e\_cparhdr);

**printf**("e\_minalloc: % 4X\n", dos\_header.e\_minalloc);

**printf**("e\_maxalloc: % 4X\n", dos\_header.e\_maxalloc);

**printf**("e\_ss: % 4X\n", dos\_header.e\_ss);

**printf**("e\_sp: % 4X\n", dos\_header.e\_sp);

**printf**("e\_csum: % 4X\n", dos\_header.e\_csum);

**printf**("e\_ip: % 4X\n", dos\_header.e\_ip);

**printf**("e\_cs: % 4X\n", dos\_header.e\_cs);

**printf**("e\_lfarlc: % 4X\n", dos\_header.e\_lfarlc);

**printf**("e\_ovno: % 4X\n", dos\_header.e\_ovno);

**printf**("e\_res[4]: % 4X\n", dos\_header.e\_res[4]);

**printf**("e\_oemid: % 4X\n", dos\_header.e\_oemid);

**printf**("e\_oeminfo: % 4X\n", dos\_header.e\_oeminfo);

**printf**("e\_res2[10]: % 4X\n", dos\_header.e\_res2[10]);

**printf**("e\_lfanew: % 4X\n", dos\_header.e\_lfanew);

    // This below code will seek to the NT header offset of an application

**fseek**(file, dos\_header.e\_lfanew, **SEEK\_SET**);

    // This below code will read the NT header signature of an application

***DWORD*** nt\_signature;

**fread**(&nt\_signature, sizeof(nt\_signature), 1, file);

    // This below code will verify the NT signature of an application

    if (nt\_signature != **IMAGE\_NT\_SIGNATURE**)

    {

**printf**("Error: Invalid NT signature\n");

**fclose**(file);

        return 1;

    }

    // This below code will read the COFF header of an application

***IMAGE\_FILE\_HEADER*** coff\_header;

**fread**(&coff\_header, sizeof(coff\_header), 1, file);

    // This below code will print out the COFF header fields of your application

**printf**("\n" **REDFONT**);

**printf**("COFF header details below:\n" **GRNFONT**);

**printf**("Machine: % 4X\n", coff\_header.Machine);

**printf**("NumberOfSections: % 4X\n", coff\_header.NumberOfSections);

**printf**("TimeDateStamp: % 4X\n", coff\_header.TimeDateStamp);

**printf**("PointerToSymbolTable: % 4X\n", coff\_header.PointerToSymbolTable);

**printf**("NumberOfSymbols: % 4X\n", coff\_header.NumberOfSymbols);

**printf**("SizeOfOptionalHeader: % 4X\n", coff\_header.SizeOfOptionalHeader);

**printf**("Characteristics: % 4X\n", coff\_header.Characteristics);

    // This below code will read the optional header

***IMAGE\_OPTIONAL\_HEADER*** optional\_header;

**fread**(&optional\_header, sizeof(optional\_header), 1, file);

    // This below code will print out the optional header fields

**printf**("\n" **REDFONT**);

**printf**("Optional header details below:\n" **GRNFONT**);

**printf**("Magic: % 4X\n", optional\_header.Magic);

**printf**("MajorLinkerVersion: % 4X\n", optional\_header.MajorLinkerVersion);

**printf**("MinorLinkerVersion: % 4X\n", optional\_header.MinorLinkerVersion);

**printf**("SizeOfCode: % 4X\n", optional\_header.SizeOfCode);

**printf**("SizeOfInitializedData: % 4X\n", optional\_header.SizeOfInitializedData);

**printf**("SizeOfUninitializedData: % 4X\n", optional\_header.SizeOfUninitializedData);

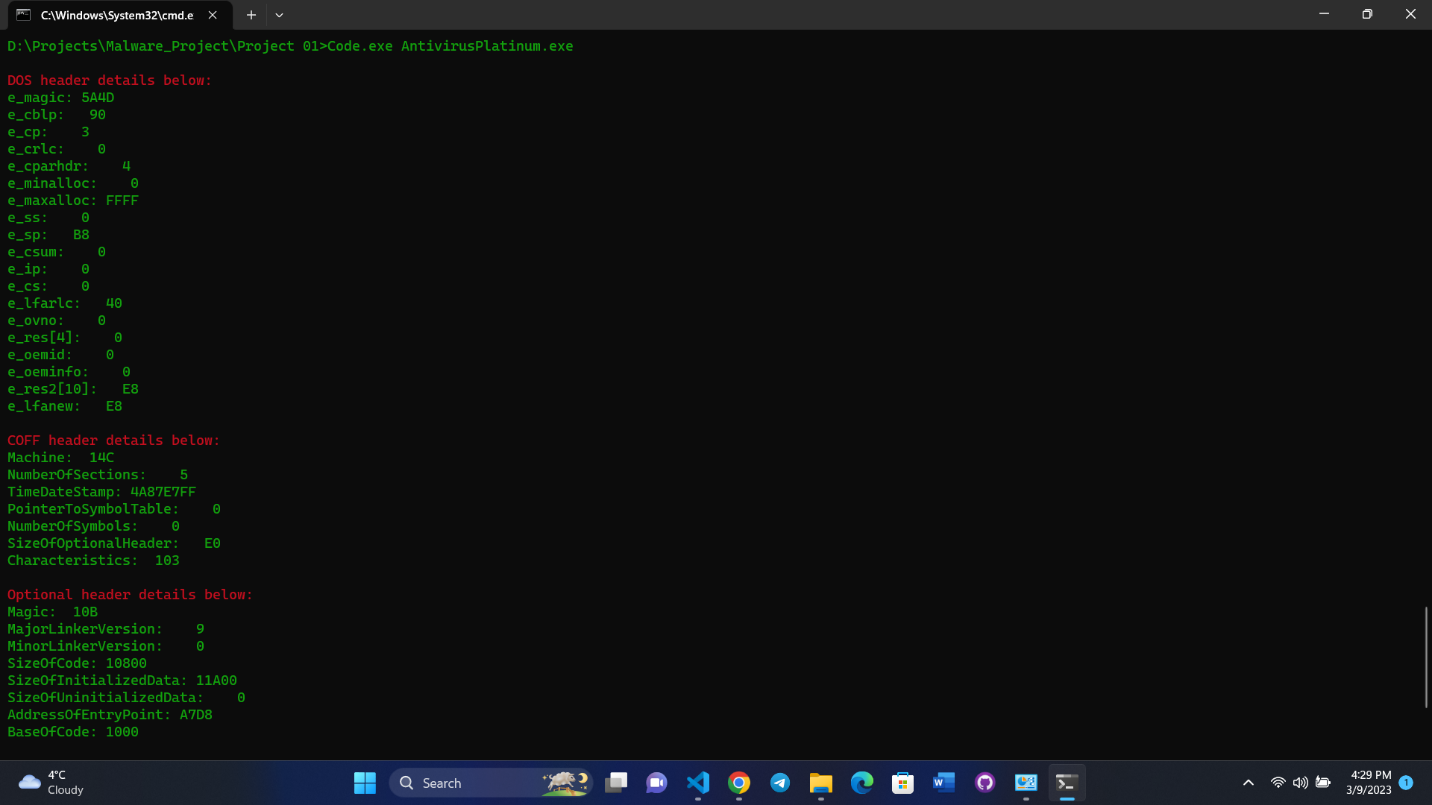
**printf**("AddressOfEntryPoint: % 4X\n", optional\_header.AddressOfEntryPoint);

**printf**("BaseOfCode: % 4X\n", optional\_header.BaseOfCode);

}

**Output**

1. Output for AntivirusPlatinum.exe



1. Output For Stardust.exe

