

**汇编语言程序设计大作业实验报告感染部分**

|  |  |
| --- | --- |
| 姓 名： | 马承乾 金为轩 李葳骏  李聪 李亚轩 连国鑫 |
| 年 级： | 2021级 |
| 指导教师： | 林颖 |
| 课程名 ： | 汇编语言程序设计 |

2023年 6 月

1. **实验目的**

1、对学生汇编语言程序实验课编程能力整体考查

2、考查学生团队合作编程的效率和团队学习能力

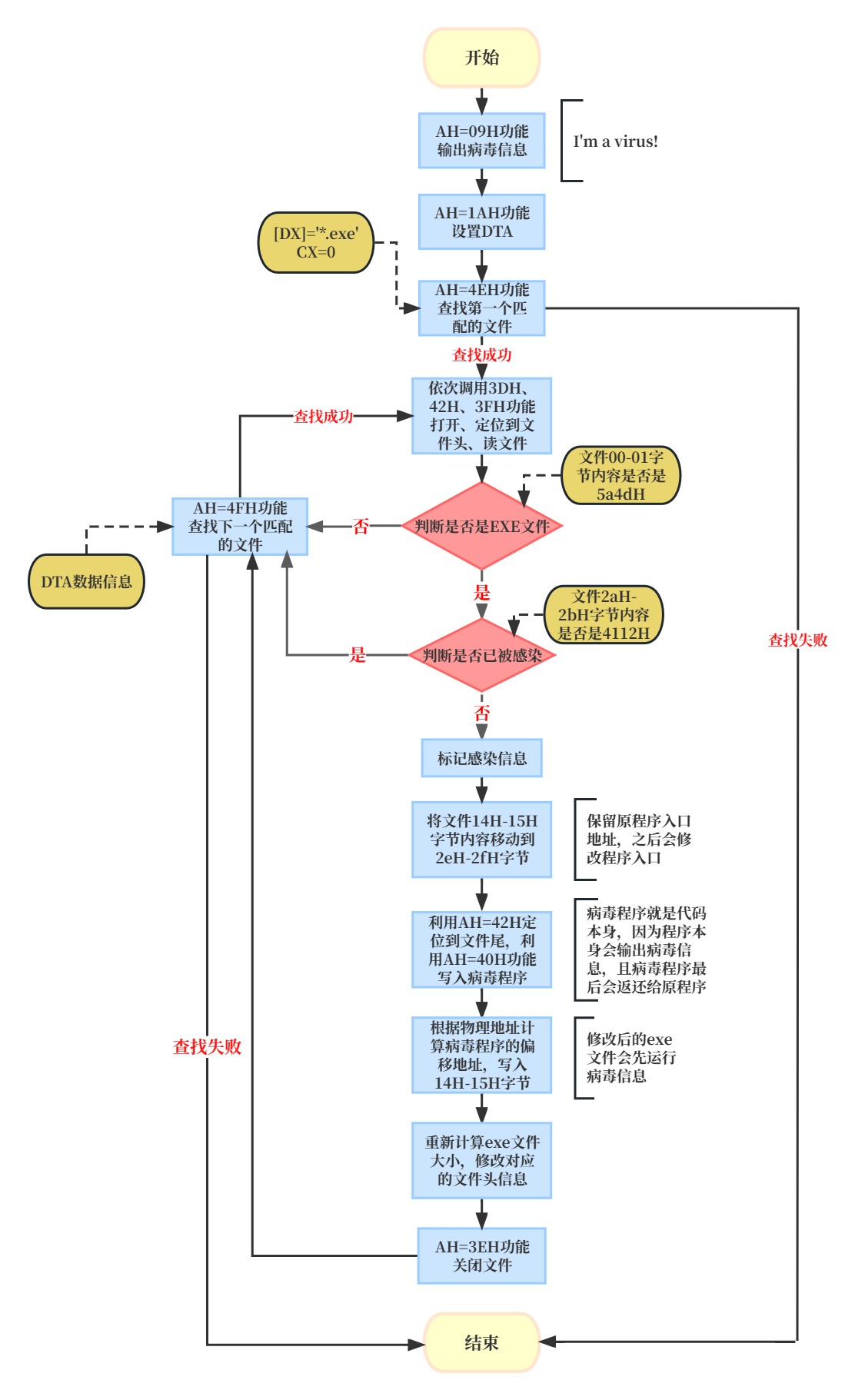
3、让学生完成具有一定的挑战性的任务

**二. 实验内容**

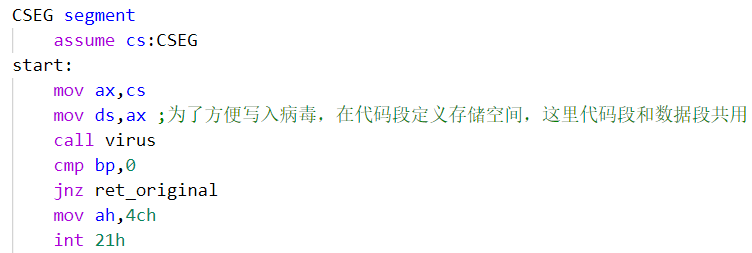
1、编写一个病毒传染程序virus.exe，运行 virus.exe能将病毒传染给DOS当前文件夹所有EXE文件，已感染病毒EXE文件不重复传染。

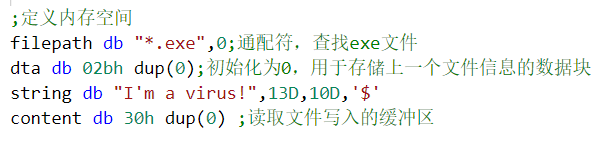
**三．实验步骤以及结果**

**1、流程图**



**2、主程序和数据定义**





为了方便写入病毒程序，所以只有代码段，关联段名与段寄存器、将段地址装入段寄存器。调用病毒子程序。BP在该代码中值为0，但在写入的exe文件中指向病毒程序的起始地址，所以根据判断BP寄存器的值来判断是否需要返回给原程序，ret\_original标签底下的代码内容就是在执行完病毒程序之后将程序返回给原程序。细节在最后面介绍。至于数据定义，定义filepath变量，存储exe文件的路径，因为感染的是同文件目录底下的exe文件，所以这里使用相对路径，使用通配符\*匹配后缀为exe的文件；dta为我们自己设置的dta地址，用于据此利用4f功能查找下一个匹配的文件，string变量存储输出的病毒字符串信息；content是读入的exe部分文件头信息，我们在这里修改，实现感染，并最后将全部修改好的内容重新写入原EXE文件。下面介绍virus子程序的具体实现。

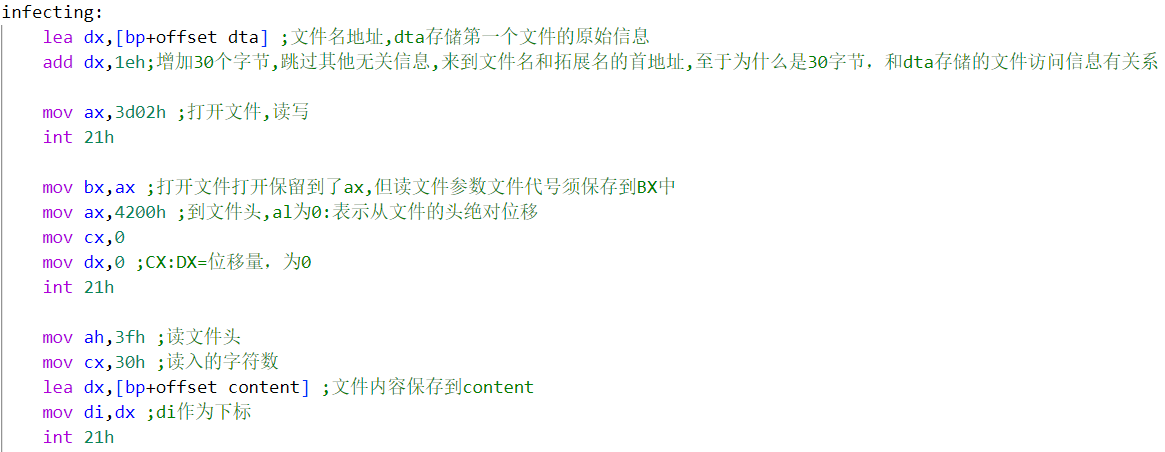
**3、virus子程序**

BP寄存器介绍、查找第一个匹配的文件



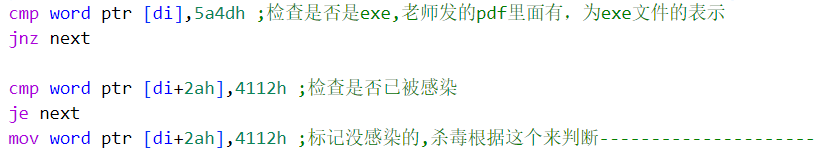
先调用标签first，将此时的IP值压入栈，再出栈返回给BP寄存器，就得到了此时的IP值，BP再减去first标签的偏移地址，这里有两种情况，一是对于代码本身，此时BP的值为0，但在将这段代码写入的EXE文件中，BP的值为病毒程序的起始地址，所以这样就可以实现上面提到的根据BP的值是否为0来判断是否需要返回给原程序。接着调用09号功能输出感染信息。再调用1aH号功能将本地的内存空间设置为DTA地址，因为4fH功能会据此查找下一个匹配的文件。最后dx寄存器载入文件路径，CX寄存器置0表示文件属性为普通文件。调用4eH号功能来查找第一个匹配的文件，在执行此操作后，dta将包含第一个匹配文件的信息，如文件名、扩展名、文件属性等。根据CF的值来判断是否查找成功：CF的值为1代表查找失败直接返回主程序，否则跳转到infecting感染部分。

感染的准备工作



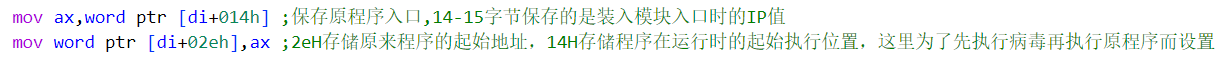
进入infecting标签中，之后利用4fH功能查找下一个匹配的文件之后会跳转到这个地方。dta存储空间中储存有当前文件的信息，dx寄存器载入此信息，并自增30个字节跳过其他无关信息,来到文件名和拓展名的首地址，再调用3dH号功能打开当前的文件，设置访问方式为读写，文件代号会保存到AX寄存器中。将其赋给BX寄存器，并调用42号功能从文件开头开始位移0，定位到文件开头，再调用3f号功能读入信息到content内存中，读取字节数为30H，为EXE重要的文件头信息，并将dx的值赋给di,使其也指向读入的文件信息的起始地址。

检测信息



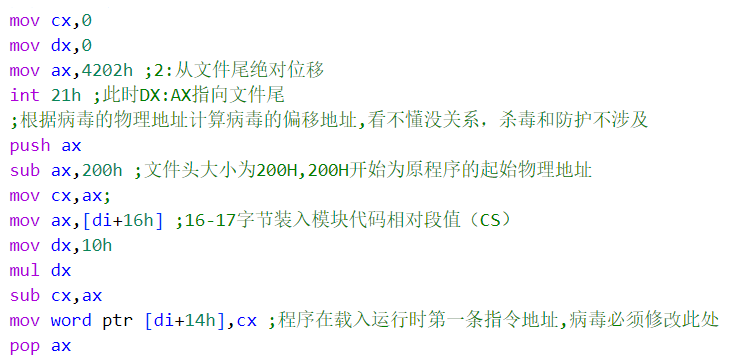
根据读入的信息的00-01字节来判断当前文件是否是exe文件，因为这个字空间存储exe文件的标准信息MZ。如果不是则跳转到匹配下一个文件。否则再判断是否已经被感染，这里因为在感染的过程中会标记exe文件的2aH-2bH字节将其设置为4112H，因此可以简单地根据这个内存空间来判断是否已经被感染。如果已经被感染也同样跳转到匹配下一个文件，避免重复感染。

保留原程序入口

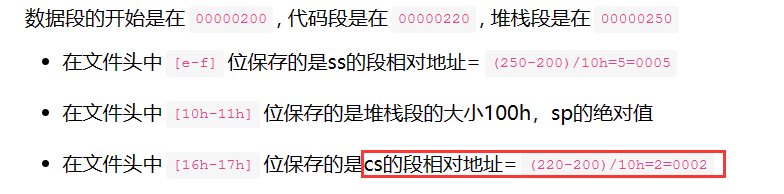


exe文件地14H-15H字节存储有程序在载入运行时第一条指令地址，即代码段IP的值，为了能够在执行完病毒程序可以返回原程序，这里需要保留此信息，我们将其移动到2eH-2fH字节保存。

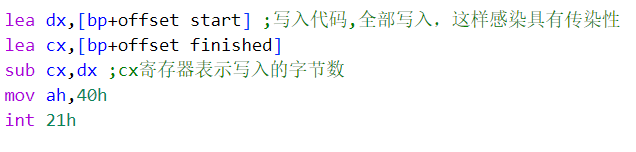
计算病毒程序的偏移地址



首先再次调用42H号功能，al设置为02表示相对于文件末尾位移。此时DX：AX为文件末尾的物理地址，我们在文件末尾写入病毒程序。接下来就是利用病毒程序的物理地址来计算偏移地址，并将其写入exe文件的14H-15H字节。物理地址值不超过两个字节，所以完全存储在AX寄存器中，将其入栈保护起来，以便后续从此处写入病毒代码，减去文件头大小200H。因为cs寄存器的值是相对文件头和数据段来的，具体可以参考下面的图片。再减去cs的值\*10H就可以得到相应的偏移地址，并将其存入14H-15H字节，覆盖掉原程序的IP值。之后再运行该exe文件就可以从病毒程序开始运行。

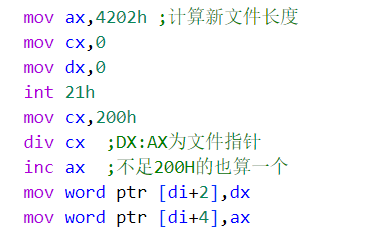


写入病毒程序



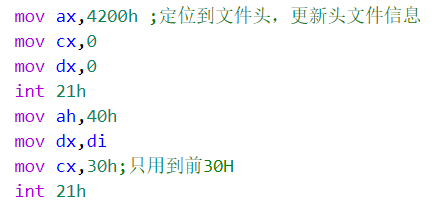
代码整个就是一个病毒程序，标记开始和结束的地方，将开始的地址载入DX寄存器，结束地址载入CX寄存器，相减得到字节数。调用40H功能将其写入当前文件指针指向的位置，即原先的文件末尾。

计算并修改新文件长度



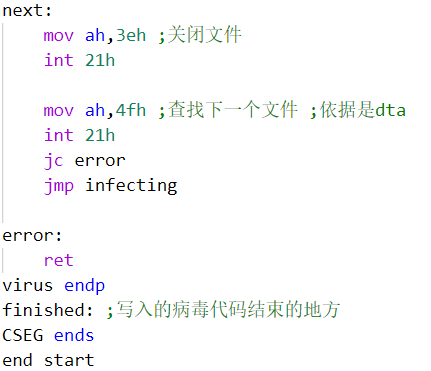
再次调用42H号功能，al设置为02表示相对于文件末尾位移。此时DX：AX为写入了病毒程序的文件末尾的物理地址，并除以200H(一个扇区的大小)，得到端格式表示当前exe在硬盘中占的扇区数量，最后一个扇区的字节数，这里要注意不足一个也算一个，所以ax自增1，并分别写入对应的文件头存储空间。

写入原文件



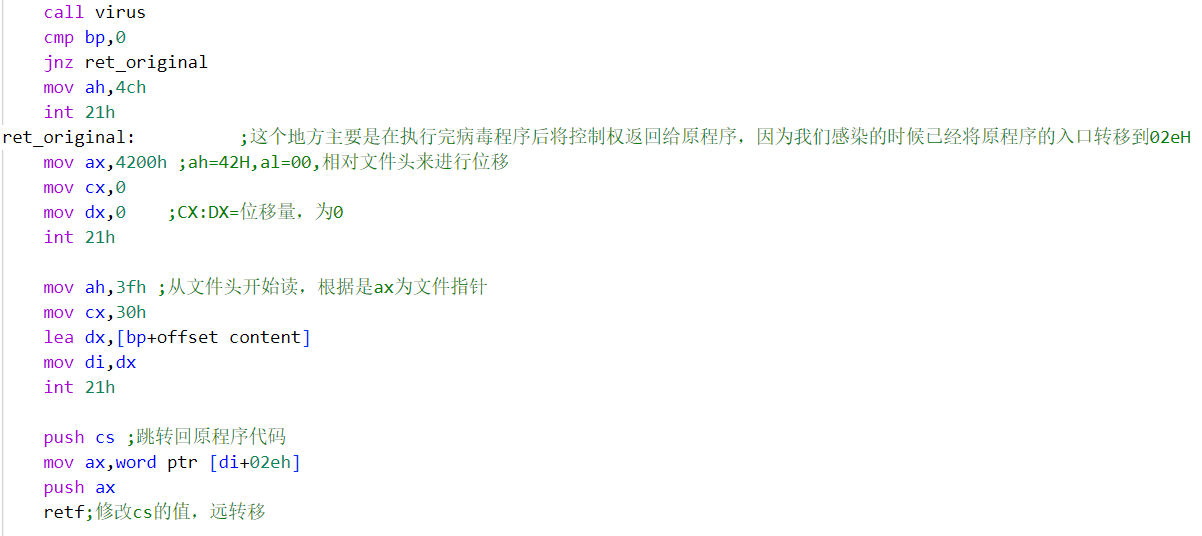
由于之前的文件头更新都是在本地的存储空间完成的，没有实际修改exe文件，所以这里定位到文件开头，写入30H个字节，覆盖掉原来的文件头信息。

查找匹配的下一个文件



关闭当前文件，根据dta信息查找下一个匹配的文件，并自动更新dta内容，存储下一个匹配文件的信息。同样判断CF的值来判断是否查找成功，如果成功跳转到infecting,否则返回主程序。

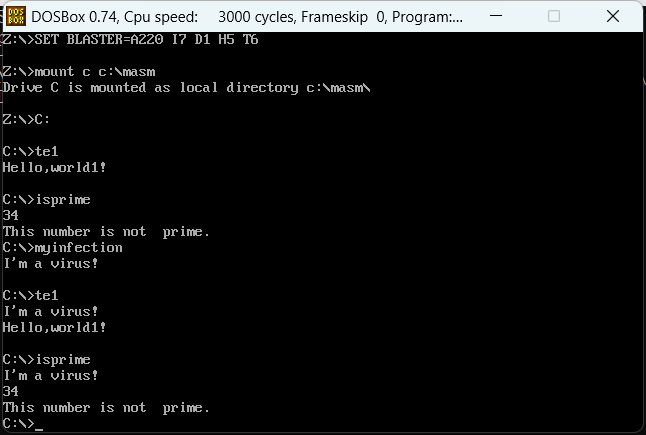
**4、在exe文件中返回原程序**

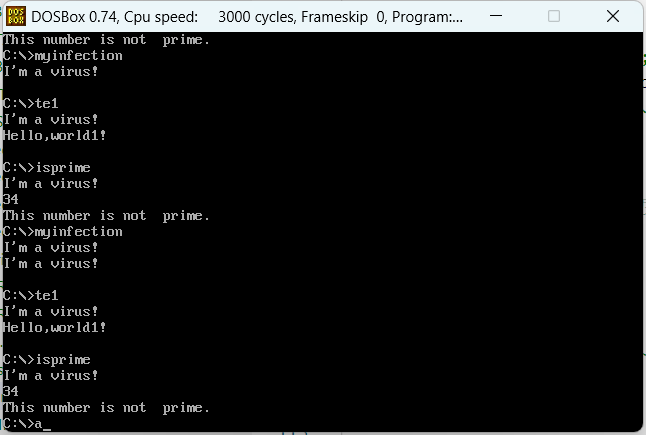


这里介绍2中提到的结束病毒程序之后返回给原程序的具体实现。之前已经介绍过了，由于BP寄存器在exe文件中指向病毒代码的起始地址，不为0，所以会跳转到ret\_original。重新定位到当前exe文件的文件起始地址，并同样读入30H个字节到content中，因为病毒程序cs的值和原程序的值相同，直接入栈，并在第2eH-2fH字节中将原程序的IP的值提取出来，同样入栈，并利用retf出栈修改CS：IP的值，就可以跳转到原程序的代码段起始地址，继续执行原程序。

**四、实验结果与分析**

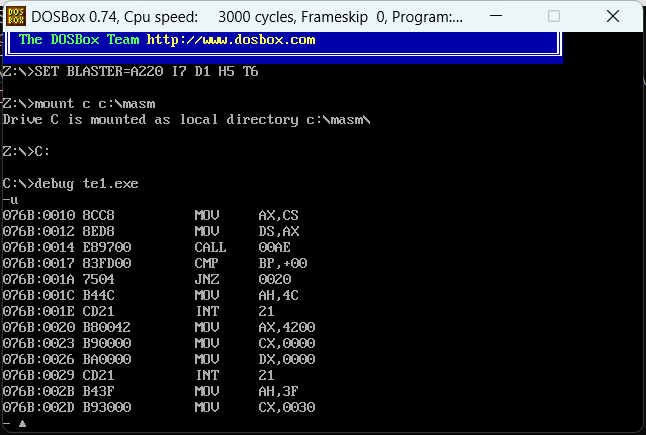
**运行效果**





感染成功，且不影响原程序的功能，也不会重复感染。

debug查看

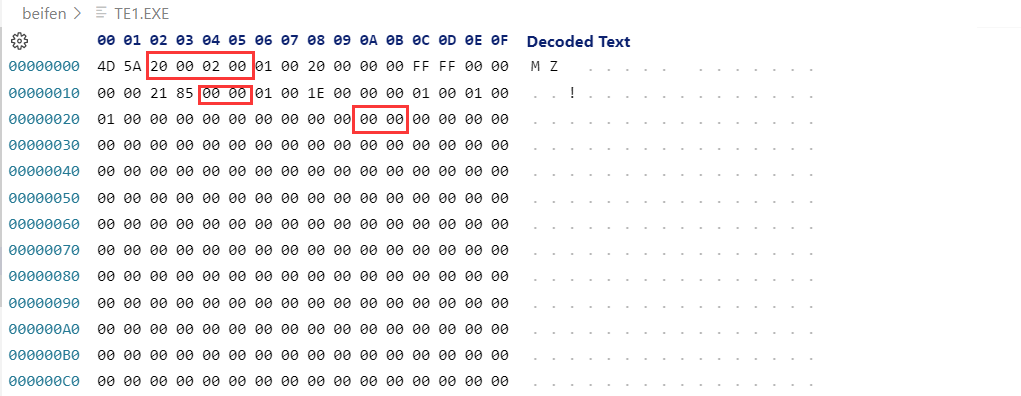


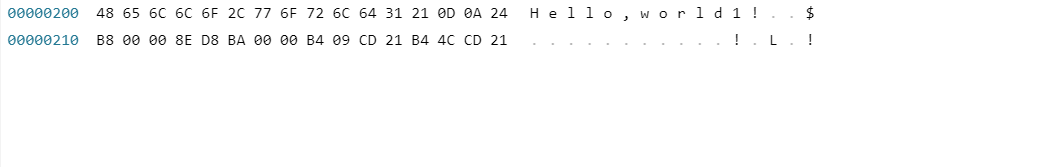


如图，进入被感染的test之后，test程序进入myinfect对应的病毒部分代码中

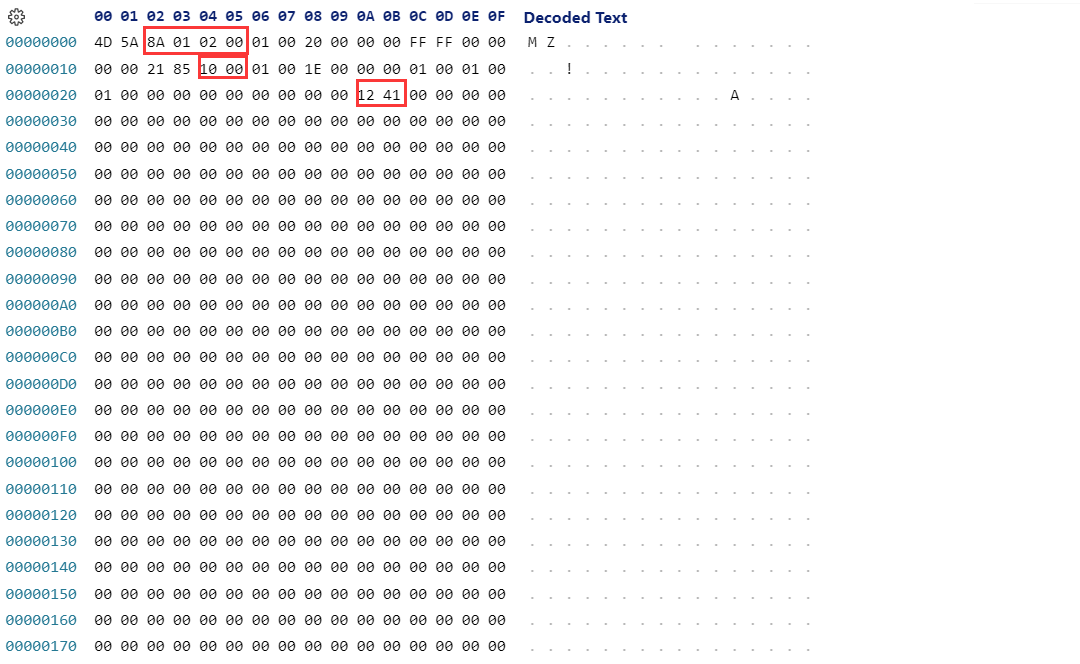
感染前后exe文件

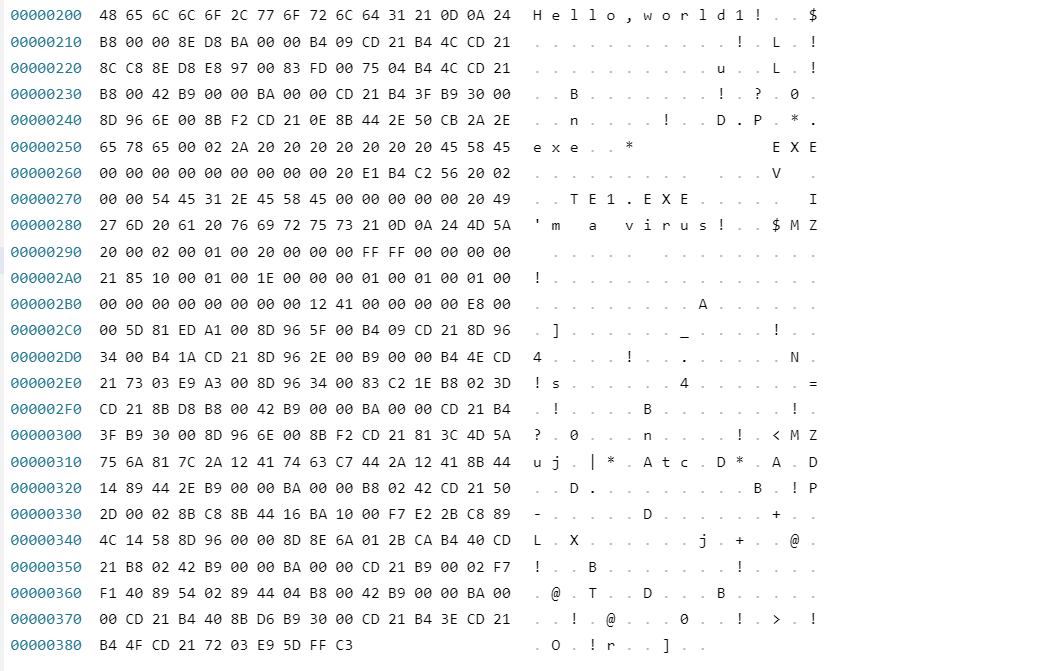
感染前





感染后





文件大小信息、感染标记信息、IP值发生了改变，病毒程序追加在原程序末尾。

**五．实验总结**

通过本次实验编写病毒感染程序:

我对可执行文件的内部结构组成更加了解：EXE可执行文件通常包含头部、代码段、数据段和其他各种元数据。通过了解这些组成部分，我能了解文件头部各字节存储的文件信息，例如文件大小等等，够理解病毒感染程序是如何将自身插入到目标文件中的：病毒可以将自己的代码添加到目标文件的代码段中，并修改头部和其他元数据，以便在执行目标文件时执行病毒代码。

我对汇编语言在文件读写操作中的应用更加熟悉了。学会了使用汇编指令来打开、读取和写入文件以及移动文件指针，定位到目标位置，通过文件操作实现病毒的传播和感染。这种对文件操作的掌握对于编写恶意软件或进行安全研究都非常重要。

我了解了DOS传统的文件查找方法中的数据传输区（Data Transfer Area，DTA）的作用：DTA是一个用于存储文件查找结果的内存区域，通过设置DTA的相关字段，可以使用DOS中的文件查找功能，获取文件的属性和路径等信息，进而对目标文件进行感染。

我学到了如何根据exe头文件中的信息计算出代码段的起始地址。在感染程序中，需要定位目标文件的代码段，以便插入恶意代码。通过分析exe头文件中对应字节信息，根据代码段入口的IP值和CS值，进而计算出代码段的起始地址，然后在相应的位置插入病毒代码。

总的来说，这次实验使我对病毒的工作原理有了更深入的理解，并且提升了我的汇编语言和文件操作技能。通过实践编写病毒感染程序，我更加意识到了安全编程的重要性，以及如何防范和检测恶意软件的方法。同时，我也认识到了在使用这些知识和技能时需要遵守法律和道德准则的重要性，以确保技术的合法和正当使用。