课程名称	操作系统	课程编号	A2130330
实验地点	综合实验楼 A511/A512	实验时间	2019-06-10
校外指导教师		校内指导教师	常光辉
实验名称	实验四 设备管理		
评阅人签字		成绩	

一、实验目的

本实验着重于了解磁盘的物理组织，以及如何通过用户态的程序直接调用磁盘 I/O API 函数（DeviceIoControl）根据输入的驱动器号读取驱动器中磁盘的基本信息，在 Windows Server 2016 环境进行。

二、工具/准备工作

1. 回顾教材相关内容；
2. 安装有 Windows Server 2016 的计算机或虚拟机；
3. 系统中装有 Visual Studio 或 Visual C++ 6.0 或其他 C++编译器。

三、实验环境

操作系统：Windows Server 2016（虚拟机）

编程语言：C++

集成开发环境：Visual Studio 2019

四、实验步骤与实验过程

步骤 1：登录进入 Windows Server 2016。

步骤 2：在“开始”菜单中单击“程序”-“Microsoft Visual Studio 2019”，进入 Visual Studio 2019 的窗口。

步骤 3：在 Visual Studio 2019 窗口的工具栏中单击“文件”按钮，在解决方案 OSLab 中新建一个项目，命名为 Lab4_1，并在源文件中添加新建项，命名为 Lab4_1.cpp，将实验指导书中的代码拷贝至该文件中。

步骤 4：单击“生成”菜单中的“编译”命令，系统对 Lab4_1.cpp 进行编译。

步骤 5：编译完成后，单击“生成”菜单中的“生成”命令，建立 Lab4_1.exe 可执行文件。

步骤 6：在解决方案资源管理器中右键当前项目，并设置为启动项目。在工具栏单

击“调试”-“开始执行（不调试）”按钮，执行 Lab4_1.exe 程序。

五、实验结果与分析

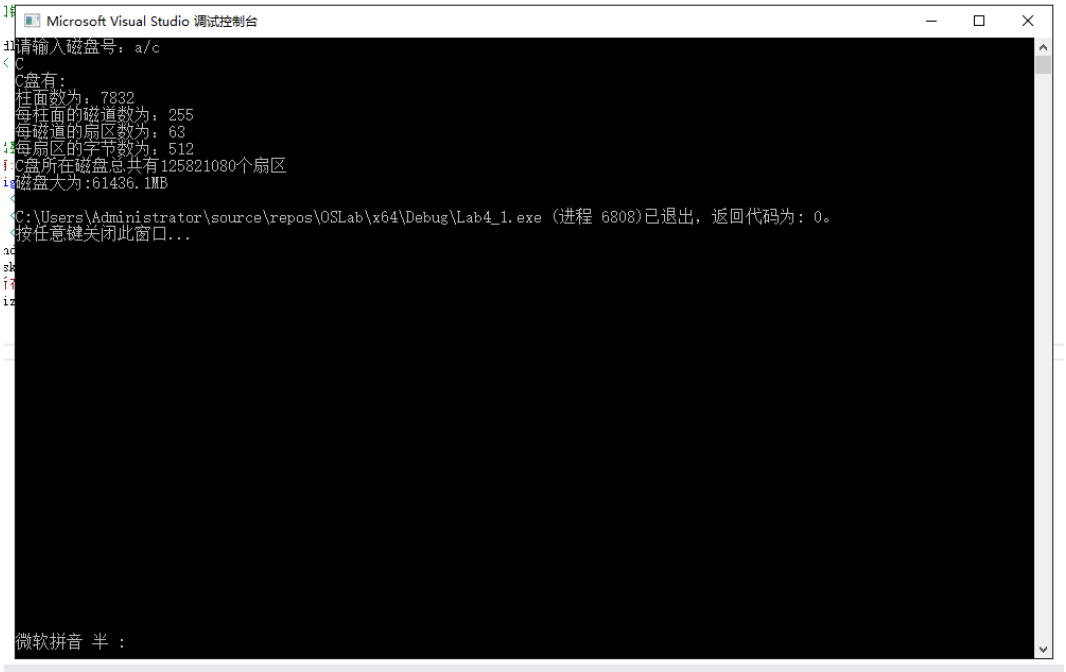

图 4-1 Lab4_1 运行结果

讨论：如输入磁盘号为 C，显示的磁盘信息是整个硬盘信息，而不是 C 盘分区的信息。如输入磁盘号为 D，显示的磁盘信息与如输入磁盘号为 C 显示的磁盘信息相同。用磁盘 I/O API 函数读出的磁盘信息是从硬盘的主引导区得到。

六、实验心得体会

通过本次实验，我了解了磁盘的物理组织，不同磁盘不同的物理构造，如 SSD 和 HDD，了解了其特点，以及如何通过用户态的程序直接调用磁盘 I/O API 函数（DeviceIoControl），使程序可以根据输入的驱动器号读取驱动器中磁盘的基本信息。

本次实验调试过程的前半段，我是使用 Windows 10 进行的，遇到了一些类型转化以及无输出的问题，纠结了很久没有弄出来，但是在 Windows Server 2016 中，没有任何问题，可以直接运行。以后调试程序应尽量在目标机器上调试，防止出现问题。

附录 程序清单

清单 4-1

```
1.  #include <windows.h>
2.  #include <iostream>
```

```
using namespace std;
#include <winioctl.h>
#include <string.h>

struct Disk //关于 Disk 结构的定义
{
    HANDLE handle;
    DISK_GEOMETRY disk_info;
};

Disk disk;
HANDLE Floppy;
static _int64 sector;
bool flag;
Disk physicDisk(char driverLetter);

void main(void)
{
    char DriverLetter;
    cout << "请输入磁盘号：a/c" << endl;
    cin >> DriverLetter;//选择要查看的磁盘
    disk = physicDisk(DriverLetter);
}

Disk physicDisk(char driverLetter) //
{
    flag = true;
    DISK_GEOMETRY* temp = new DISK_GEOMETRY;
    char device[9] = "\\\\.\\c:";
    device[4] = driverLetter;
    Floppy = CreateFile(device,                    //将要打开的驱动器名
      GENERIC_READ,              //存取的权限
      FILE_SHARE_READ | FILE_SHARE_WRITE, // 共享的权限
      NULL,                      //默认属性位
      OPEN_EXISTING,             //创建驱动器的方式
      0,                       //所创建的驱动器的属性
      NULL);                     //指向模板文件的句柄
    if (GetLastError() == ERROR_ALREADY_EXISTS)   //如打开失败，返回错误代码
    {
      cout << "不能打开磁盘" << endl;
      cout << GetLastError() << endl;
      flag = false;
      return disk;
    }

    DWORD bytereturned;
```

```
49.     BOOL Result;
50.     disk.handle = Floppy;
51.     Result = DeviceIoControl(Floppy,
52.       IOCTL_DISK_GET_DRIVE_GEOMETRY,
53.       NULL,
54.       0,
55.       temp,
56.       sizeof(*temp),
57.       &bytereturned,
58.       (LPOVERLAPPED)NULL);
59.     if (!Result) //如果失败，返回错误代码
60.     {
61.       cout << "打开失败" << endl;
62.       cout << "错误代码为：" << GetLastError() << endl;
63.       flag = false;
64.       return disk;
65.     }
66.
67.     disk.disk_info = *temp;//输出整个物理磁盘的信息
68.     cout << driverLetter << "盘有: " << endl;
69.     cout << "柱面数为：" << (unsigned long)disk.disk_info.Cylinders.QuadPart << endl;
70.     cout << "每柱面的磁道数为：" << disk.disk_info.TracksPerCylinder << endl;
71.     cout << "每磁道的扇区数为：" << disk.disk_info.SectorsPerTrack << endl;
72.     cout << "每扇区的字节数为：" << disk.disk_info.BytesPerSector << endl;
73.     sector = disk.disk_info.Cylinders.QuadPart * (disk.disk_info.TracksPerCylinder) * (disk.disk_info.SectorsPerTrack);
74.     double DiskSize = (double)disk.disk_info.Cylinders.QuadPart * (disk.disk_info.TracksPerCylinder) * (disk.disk_info.SectorsPerTrack) * (disk.disk_info.BytesPerSector);
75.     cout << driverLetter << "盘所在磁盘总共有" << (long)sector << "个扇区" << endl;
76.     cout << "磁盘大为:" << DiskSize / (1024 * 1024) << "MB " << endl;
77.     delete temp;
78.     return disk;
79.   }
```