问题清单

1. boot.asm 文件中，org 0700h 的作用

告诉汇编器该段代码会被加载到内存的07c00 处，当编译的时候遇到

相对寻址的指令的时候会用07c00 加上相对地址得到绝对地址，

2. 为什么要把boot.bin 放在第一个扇区？直接复制为什么不行？

BIOS 程序检查软盘0 面0 磁道1 扇区，如果扇区以0xaa55 结束，

则认定为引导扇区，将其512 字节的数据加载到内存的07c00 处，然后设

置PC，跳到内存07c00 处开始执行代码。

普通的读写操作（mv, rm,cp）是基于文件系统的，文件系统是一个

逻辑概念。而引导扇区，是磁盘第一个磁道的第一个扇区，他是一个物理

概念，在文件系统中，这个扇区是不可见的。

3. loader 的作用有哪些?

加载内核入内存，跳入保护模式，内存分页。

1

4. 说明下面每行代码的意思。

行号代码

```
1 L1 db 0
2 L6 dd 1A92h
3 mov al, [L1]
4 mov eax, L1
5 mov [l1], ah
6 mov eax, [L6]
7 add eax, [L6]
8 add [L6], eax
9 mov al, [L6]
```

L1 db 0; 字节变量L1，初始值为0

L6 dd 1A92h ; 双字变量初始化成十六进制1A92

3 mov al, [L1] ; 复制L1 里的字节数据到AL

4 mov eax, L1 ; EAX = 字节变量L1 代表的地址
5 mov [L1], ah ; 把AH 拷贝到字节变量L1

6 mov eax, [L6] ; 复制L6 里的双字数据到EAX

7 add eax, [L6] ; EAX = EAX + L6 里的双字数据

8 add [L6], eax ; L6 = L6 里的双字数据+ EAX

9 mov al, [L6] ; 拷贝L6 里的数据的第一个字节到AL

注:
word(字) 2 个字节
double word(双字) 4 个字节
quad word(四字) 8 个字节

paragraph(一节) 16 个字节

5. times 510-($-$$) db 0

为什么是510? $ 和$$ 分别表示什么? 不用times 指令怎么写(等价命

令)?
因为需要填充512 个字节的数据,最后两个字节是以0xaa55 结尾,

所以需要填充510 个字节$ 表示当前的字节数,$$ 表示开始的字节。

2

不用times 命令可以使用db 0 循环($-$$) 次

6. 解释db 命令: L10 db "w", "o", "r", "d",0 这条语句的意

义,并且说明数字0 的作用。

填充字符串"word",最后的0 表示结束符,即在C/C++ 里字符串末

尾的结束符。
备注:

这个答案也是大多数同学的答案,可以说正确,也可以说不正确,因

为是依赖于场景的,这道题目来源于参考资料PCASM 一书,那本书中调

用了C 库函数,所以注释中是上文答案。而0 其实并不是汇编语言中的

结束符,汇编里面的结束符是$ 等符号(根据平台有所不同)表示。(感谢

131250012 同学提出这个问题,并写程序实验验证,他将得到额外的加分,
如果其他同学在以后的实验中有类似性质的实验验证,也将得到奖励)。
7. L1 db 0

L2 dw 1000

L1、L2 是连续存储的吗？即是否L2 就存储在L1 之后？

连续定义的数据储存在连续的内存中。也就是说，字L2 就储

存在L1 的后面。

8. 要是不知道L6 标识的是多大的数据，下面这句话对不对？

mov [L6], 1

This statement produces an operation size not specified error. Why? Because the assembler does not know whether to store the 1 as a byte, word or double word. To fix this, add a size specifier: mov dword [L6], 1 ; store a 1 at L6 This tells the assembler to store an 1 at the double word that starts at L6. Other size specifiers are: BYTE, WORD, QWORD and TWORD.

9. 如何处理输入输出？在代码中哪里体现出来？

使用中断处理。

10. 通过什么来保存前一次的运算结果？在代码中哪里体现出来？

栈或者寄存器

11. 随机选择代码段，说明作用。

12. 有哪些段寄存器？

代码段，数据段，堆栈段，附加段

13. 8086/8088 存储单元的物理地址长是多少？地址总线有多少位？可以直接寻址的物理空间是多少？

分别是20、20、1M。

说明：8086/8088 的地址总线都是20 位的；外部数据总线宽度：8086：16 位；8088：8 位；内部数据总线宽度相同，都是16 位

物理地址= 段值*16+ 偏移（左移四位）（实模式下如此，其他模式下16 会变化）

14. 寄存器的寻址方式（知道如何计算）。

立即寻址方式；寄存器寻址方式；直接寻址方式；寄存器间接寻址方

式；寄存器相对寻址方式；基址加变址寻址方式；相对基址加变址寻址方

式

15. 几个常用指令的作用（如MOV，LEA 等）。

MOV: 把一个字或字节从源操作数SRC 送至目的操作数DST

LEA: 把操作数OPRD 的有效地址传送到操作数REG

PUSH、POP：堆栈操作指令
ADD、ADC、SUB、SBB：加减运算指令
——其他见PPT，更详细的请翻阅《80X86》

16. 主程序与子程序的几种参数传递方式。

利用寄存器传递参数

利用约定存储单元传递参数

利用堆栈传递参数

利用CALL 后续区传递参数