最简操作系统

项目网址:https://os.phil-opp.com/multiboot-kernel/#Linker%201M
链接文件编写规则教程网站：
https://sourceware.org/binutils/docs/ld/Basic-Script-Concepts.html

综述：使用 ld 文件里的编译规则编译 multiboot_header.asm　　boot.asm 文件产生 kernel.bin 文件,并编写 grub.cfg 文件 ，用目录里的方式组织文件，最后再用相应的命令编译项目文件夹以此来产生 os.iso 系统镜像文件。

Hint:具体的命令和文件为何这么些，请查阅项目网址。

1.项目目录：

```
isofiles
└─── boot
     ├─── grub
     │    └─── grub.cfg
     └─── kernel.bin
```

2.grub.cfg 文件：

```
set timeout=0
set default=0

menuentry "my os" {
    multiboot2 /boot/kernel.bin
    boot
}
```

3.产生 kernel.bin 文件的过程：
共依赖三个文件：`linker.ld`　`multiboot_header.asm`　`boot.asm`
4.文件内容：
1). `linker.ld:`

```
ENTRY(start)

SECTIONS {
    . = 1M;

    .boot :
    {
        /* ensure that the multiboot header is at the beginning */
        *(.multiboot_header)
    }

    .text :
```

```
    {
        *(.text)
    }
}
```

2). multiboot_header.asm:

```
section .multiboot_header
header_start:
    dd 0xe85250d6                ; magic number (multiboot 2)
    dd 0                         ; architecture 0 (protected mode i386)
    dd header_end - header_start ; header length
    ; checksum
    dd 0x100000000 - (0xe85250d6 + 0 + (header_end - header_start))

    ; insert optional multiboot tags here

    ; required end tag
    dw 0    ; type
    dw 0    ; flags
    dd 8    ; size
header_end:
```

3). boot.asm:

```
global start

section .text
bits 32
start:
    ; print `OK` to screen
    mov dword [0xb8000], 0x2f4b2f4f
    hlt
```

5.使用的命令：

```
nasm -f elf64 multiboot_header.asm//产生二中间文件.o
nasm -f elf64 boot.asm
ld -n -o kernel.bin -T linker.ld multiboot_header.o boot.o
sudo apt-get install xorriso
```

```
grub-mkrescue -o os.iso isofiles//在项目目录文件夹所在的文件夹里执行
```

说明：执行以上三个命令后，会产生 kernel.bin 文件,也就是项目目录里的对应文件

备注：如果想在 qemu 里运行该系统，应使用命令（前提是要安装 qemu 的对应硬件模拟环境，此处为 i386 的 cpu 架构：

```
qemu-system-x86_64 -cdrom os.iso
```