CpS 230 Homework 4: Linking

Specially Prepared for Mr. J (jpjuecks)

Linking

You will "link" (by hand) 2 provided object files *(see final page)* into a single binary image that will be loaded/executed at address **0x6a800** in memory.

1. Concatenate Sections *(5 pts)*

Fill in the following table indicating where in memory each section from each object file will be located. Enter both the *relative offset* (starting at 0) and the *loaded address* (i.e., $base + offset$). Align all sections on 16-byte boundaries (i.e., all the starting addresses should end in "`0`" in hex).

Module/Section	Relative Offset	Loaded Address
`hen.text`		
`jackal.text`		
`hen.data`		
`jackal.data`		

2. Resolve Symbols *(10 pts)*

Indicate the name, source section, offset in source section, and final loaded address of each public symbol, in order of final loaded address.

Symbol	From	Offset	Loaded Address

3. Apply Relocations *(15 pts)*

For each relocation (in order of "site"), indicate the source section, offset in source section, final loaded address ("site"), target symbol name, original (pre-fixup) 32-bit hex value, and adjusted (post-fixup) 32-bit hex value.

Section	Offset	Site	Target	Kind	Orignal Value	Adjusted Value

4. Generate Final Image *(10 pts)*

Using a *hex editor* of your choice, construct the sequence of bytes produced by linking the given given object files, saving it as `image.bin` and submitting it electronically. `image.bin` should be exactly 94 bytes long and should have an MD5 checksum of `91bd27e0d9d8ea26ed65e4f50e7a7d4f`.

Source Files

For your reading pleasure, see the NASM source files from which your object files were assembled. (Just as a real-life linker never looks at the original source code, you don't *need* to see these files to complete this assignment. But comparing the original source to the resulting object file is an enlightening experience. :-))

"jackal.asm"

```
extern _drill
extern _level

section .text
    int3
    int3
    int3
    int3
    int3

global _hammer
_hammer:
    push    ebp
    mov     ebp, esp

    push    dword [fire_engine]
    call    _drill
    add     esp, 4

    pop     ebp
    ret

section .data
    db 0,0
fire_engine  dd  0xfa
    db 0,0,0,0
global scooter
scooter  dd  0x1c2
```

"hen.asm"

```
extern _hammer
extern scooter

section .text
    int3
    int3
    int3

global _drill
_drill:
    mov eax, [lemon]
    xor eax, [scooter]
    ret

    int3
    int3

global _level
_level:
    mov eax, [scooter]
    add eax, [pomegranate]
    ret


section .data
pomegranate  dd  0x3b6
lemon  dd  _hammer
```

Object Files

Module “jackal.obj”

.text Section Bytes/Relocations

```
00000000: CC CC CC CC CC 55 89 E5  FF 35 02 00 00 00 E8 00  .....U...5......
00000010: 00 00 00 83 C4 04 5D C3                          ......].
```

Offset	Kind	Target Symbol
10	DIR32	jackal.data
15	REL32	_drill

.data Section Bytes/Relocations

```
00000000: 00 00 FA 00 00 00 00 00  00 00 C2 01 00 00        ..............
```

Offset	Kind	Target Symbol
(no relocations for this section)		

Public/External Symbols

Section	Offset	Name
<external>	0	_drill
<external>	0	_level
jackal.text	5	_hammer
jackal.data	10	scooter

Module “hen.obj”

.text Section Bytes/Relocations

```
00000000: CC CC CC A1 04 00 00 00  33 05 00 00 00 00 C3 CC  ........3.......
00000010: CC A1 00 00 00 00 03 05  00 00 00 00 C3           .............
```

Offset	Kind	Target Symbol
4	DIR32	hen.data
10	DIR32	scooter
18	DIR32	scooter
24	DIR32	hen.data

.data Section Bytes/Relocations

```
00000000: B6 03 00 00 00 00 00 00                          ........
```

Offset	Kind	Target Symbol
4	DIR32	_hammer

Public/External Symbols

Section	Offset	Name
<external>	0	_hammer
<external>	0	scooter
hen.text	3	_drill
hen.text	17	_level