header 1/2
Technical details for identification and execution

ELF header
identify as an ELF type
specify the architecture

Program Header table
Execution information

sections
Contents of the executable

Code
Executable information

Data
Information used by the code

Sections' names

Section Header table
Linking (connecting program objects) information

header 2/2
Technical details for linking
(ignored for execution)

Hexadecimal dump | ASCII dump

```
7F 45 4C 46 01 01 01 00 00 00 00 00 00 00 00 00    .ELF............
02 00 03 00 01 00 00 00 60 00 00 08 40 00 00 00    ........`...@...
C0 00 00 00 00 00 00 00 34 00 20 00 01 00 28 00    +.......4.....(.
04 00 03 00                                        ....
```

1 Fields | Values | Explanation

Fields	Values	Explanation
e_ident		
EI_MAG	0x7F, "ELF"	constant signature
EI_CLASS, EI_DATA	1 ELFCLASS32, 1 ELFDATA2LSB	32 bits, Little-Endian
EI_VERSION	1 EV_CURRENT	Always 1
e_type	2 ET_EXEC	Executable
e_machine	3 EM_386	Intel 386 (and later)
e_version	1 EV_CURRENT	Always 1
e_entry	0x8000060	Address where execution starts **3**
e_phoff	0x40	Program Headers' offset
e_shoff	0xC0	Section Headers' offset
e_ehsize	0x34	Elf header's size
e_phentsize	0x20	Size of a single Program Header
e_phnum	1	Count of Program Headers
e_shentsize	0x28	Size of a single Section Header
e_shnum	4	Count of Section Headers
e_shstrndx	3*	Index of the names' section in the table

```
Offset:0x40/Address:0x8000040
01 00 00 00 00 00 00 00 00 00 00 08 00 00 00 08    ................
A0 00 00 00 A0 00 00 00 05 00 00 00 00 00 00 00    ................
```

2 Fields | Values | Explanation

Fields	Values	Explanation
p_type	1 PT_LOAD	The segment should be loaded in memory
p_offset	0	Offset where it should be read
p_vaddr	0x8000000	Virtual address where it should be loaded
p_paddr	0x8000000	Physical address where it should be loaded
p_filesz	0xA0	Size on file
p_memsz	0xA0	Size in memory
p_flags	5 PF_R|PF_X	Readable and eXecutable

---

3 x86 assembly | Equivalent C code

```
Offset:0x60/Address:0x8000060
¹B9 90 00 00 08 ²BA 0D 00 00 00 ³BB 01 00 00 00 ⁴B8    ................
04 00 00 00 ⁵CD 80 ⁶BB 01 00 00 00 ⁷B8 01 00 00 00    ................
⁸CD 80                                               ..
```

```asm
mov ecx, 0x8000090  MSG
mov edx, 0xD        MSG_LEN
mov ebx, 1          STDOUT_FILENO
mov eax, 4          SC_WRITE
int 0x80            system call        → write(STDOUT_FILENO, "Hello World!\n", len("Hello World!\n"));

mov ebx, 1          return code
mov eax, 1          SC_EXIT
int 0x80            system call        → exit(1);
```

```
Offset:0x90/Address:0x8000090
48 65 6C 6C 6F 20 57 6F 72 6C 64 21 0A 00    Hello.World!..
```

Strings

"Hello World!\n", 0

```
Offset:0xA0
00 2E 73 68 73 74 72 74 61 62 00 2E 74 65 78 74    ..shstrtab..text
00 2E 72 6F 64 61 74 61 00                          ..rodata.
```

Section names

"" .shrtrtab .text .rodata

```
Offset:0xC0
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 0B 00 00 00 01 00 00 00    ................
06 00 00 00 60 00 00 08 60 00 00 00 22 00 00 00    ....`...`..."...
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
11 00 00 00 01 00 00 00 02 00 00 00 90 00 00 08    ................
90 00 00 00 0D 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 01 00 00 00 03 00 00 00    ................
00 00 00 00 00 00 00 00 A0 00 00 00 19 00 00 00    ................
```

Section Header table

Index	Name (sh_name)	Type (sh_type)	Flags (sh_flags)	Address (sh_addr)	Offset (sh_offset)	Size (sh_size)
0	<null>	0 SHT_NULL (inactive)				
1	.text	1 SHT_PROGBITS (program)	6 SHF_ALLOC (allocated) SHF_EXECINSTR (executable)	0x8000060	0x60	0x22
2	.rodata	1 SHT_PROGBITS (program)	2 SHF_ALLOC (allocated)	0x8000090	0x90	0x0D
3*	.shrtrtab	3 SHT_STRTAB (string table)			0xA0	0x19

relative offsets
in names' section

This is the whole file, however, most ELF files contain many more elements.
Explanations are simplified, for conciseness.