4.

符号	是否在SWAP.O的符号表中	定义模块	符号类型	节
buf	在	main.o	extern	.data
bufp0	在	swap.o	global	.data
bufp1	在	swap.o	local	.bss
incr	在	swap.o	local	.text
count	在	swap.o	local	.data
swap	在	swap.o	global	.text
temp	不在	-	-	-

5.

（1）在上述两个文件中,main.c 中的强符号有 x、z、main,弱符号有 y;proc1.c 中的强符号有 proc1， 弱符号有 x。其中，符号 x 在 main.c 和 proc1.c 中都有定义，属于多重定义符号。根据多重定义 符号处理规则 2（若一个符号被说明为一次强符号定义和多次弱符号定义，则按强符号定义为 准），符号 x 的定义以 main.c 中的强符号 x 为准，即在 main.o 的.data 节中分配 x，占 4 个字节， 随后两个字节存放 y，再后面两个字节存放 z。

（2）程序执行时，在调用 proc1()函数之前，&x 中存放的是 x 的机器数：00000101H，随后两个字节 （地址为&y）存放 y，它没有初始化，故通常为 0；再后面两个字节存放 z 的机器数：0002H。 如图所示：

	0	1	2	3
&y	00	00	02	00
&x	01	01	00	00

	0	1	2	3
&y	00	00	F8	BF
&x	00	00	00	00

在调用 proc1()函数以后，因为 proc1()中的符号 x 是弱符号，因此，x 的定义以 main 中的强符号 x 为准，执行 x=-1.5 后，便将“-1.5”的机器数 BFF80000 00000000H 存放到了&x 开始的 8 个 字节中。即&x 中为其低 32 位的 00000000H，&y 中为高 32 位的 BFF80000H 中的低 16 位 0000H， &z 中为高 32 位的 BFF80000H 中的高 16 位 BFF8H。如上右图所示。 因此，最终打印的结果如下：x=0, z=-16392。

（3）只要将文件 proc1.c 中的第 1 行修改为“static double x;”就可以使得 proc1 中的 x 设定为本地变 量,从而在 proc1.o 的.data 节中专门分配存放 x 的 8 个字节空间，而不会和 main 中的 x 共用同 一个存储地址。因此，也就不会破坏 main 中 x 和 z 的值。

7.

全局符号 main 在 m1 中是强符号，在 m2 中弱符号，因此在 m1 中全局符号 main 被定义在.text 节中，因为 main 函数对应的机器码开始两个字节为 55H 和 89H；在 m2 中的 printf 语句中引用数组元素 main[0] 和 main[1] 时，main[0] = 55H, main[1] = 89H

8.

.data 节中全局变量的初始值总的数据长度为 0xE8，因此，虚拟地址空间中长度为 0x104 字节的可读写数据段中，开始的 0xE8 个字节取自 .data 节，后面的 28 个字节是未初始化全局变量所在区域.

9.

（1）gcc -static -o p p.o libx.a liby.a

（2）gcc -static -o p p.o libx.a liby.a libx.a

（3）gcc -static -o p p.o libx.a liby.a libz.a libx.a

10.

main.o 的.text 节中只有一个符号需要重定位，它就是在 main.c 中被引用的全局符 号 swap；需要重定位的是图 4.15 中第 6 行 call 指令中的偏移量字段，其位置相对于.text 节起始位置位移 量为 7，按照 PC 相对地址方式（R_386_PC32）。

重定位前，在位移量 7、8、9、a 处的初始值 init 的内容分别为 fc ff ff ff，其机器数为 0xfffffffc，值为 -4。重定位后，应该使 call 指令的目标转移地址指向 swap 函数的起始地址。

main 函数总共占 12H=18 字节的存储空间，其起始地址为 0x8048386，因此，main 函数最后一条指令地址为：0x8048386+0x12=0x8048398。

因为 swap 函数代码紧跟在 main 后且首地址按 4 字节边界对齐，故 swap 的起始地址就是 0x8048398。

重定位值的计算公式为：ADDR(r_sym) -((ADDR(.text)+r_offset)-init) = 0x8048398-((0x8048386+7)-(-4)) = 7

因此，重定位后，在位移量 7、8、9、a 处的 call 指令的偏移量字段为 07 00 00 00。

11.

序号	符号	位移	指令所在行号	重定位类型	重定位前内容	重定位后内容
1	bufp1（.bss）	0x8	6~7	R_386_32	0x00000000	0x8049620
2	buf（.data）	0xc	6~7	R_386_32	0x00000004	0x80495cc
3	bufp0（.data）	0x11	10	R_386_32	0x00000000	0x80495d0
4	bufp0（.data）	0x1b	14	R_386_32	0x00000000	0x80495d0
5	bufp1（.bss）	0x21	17	R_386_32	0x00000000	0x8049620
6	bufp1（.bss）	0x2a	21	R_386_32	0x00000000	0x8049620