5.

（1）

main.c 中的强符号有 z,x,main，弱符号有 y

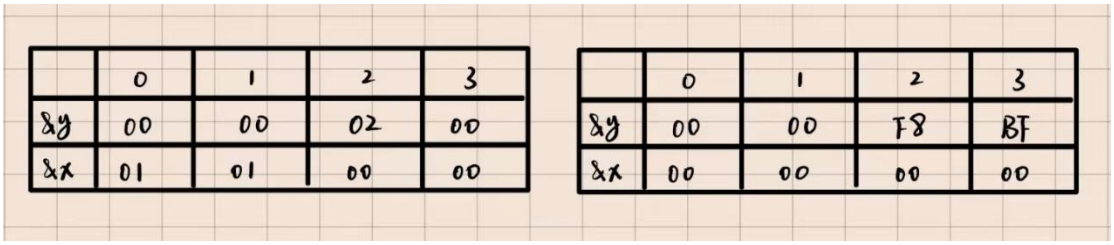
proc1.c 中的强符号有 proc1，弱符号有 x

x 以强符号定义为准。

（2）

打印结果为 x=0,z=-16392

存放内容为



	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

（3）

```
static double x;
void proc1() {
    x=-1.5;
}
```

7.

因为全局符号 main 在 m1 中是强符号，而在 m2 中是弱符号，但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，因此，以 m1 中 main 的定义为准，为强符号。所以在这里会打印出 main 函数地址后的两条机器码。

8.

据图可知，.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 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.

swap 需要进行重定位。

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

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 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。