Homework 9

Problem 1

```
/* swap.c */
extern int buf[];

int *bufp0 = &buf[0];
int *bufp1;

static void incr()
{
    static int count = 0;
    count++;
}

void swap()
{
    int temp;
    incr();

    bufp1 = &buf[1];
    temp = *bufp0;
    *bufp0 = *bufp1;
    *bufp1 = temp;
}
```

Suppose swap.o is generated according to swap.c shown above. For each symbol that is defined or referred in swap.c, indicate whether there exists an entry in symbol table of swap.o (**Y or N**). If so, indicate the object file where the symbol is defined (**swap.o or main.o**), type of symbol (**extern or global**), and section of symbol (**.text, .data, or .bss**).

Symbol	In swap.o.symtab ?	Type	Object file	Section
buf	Y	extern	main.o	.data
bufp0	Y	global	swap.o	.data
bufp1	Y	global	swap.o	.bss
swap	Y	global	swap.o	.text
temp	N	-----	-----	-----
incr	Y	local	swap.o	.text
count	Y	local	swap.o	.data(or .bss)

Problem 2

```
/* foo5.c */
#include <stdio.h>
void f(void);

int x = 15213;
int y = 15212;

int main()
{
    f();
    printf("x= 0x%x y = 0x%x \n", x, y);
    return 0;
}

/* bar5.c */
double x;
void f()
{
    x = -0.0;
}
```

As is mentioned in class, modification to *x* in bar5.c will overwrite *x* and *y* in foo5.c. Try to modify bar5.c without changing any variable name, and let foo5.c print the correct result.
(Hint: localization of *x* in bar5.c may take effect.)

```
/* bar5.c */
static double x;
……
```

Problem 3

The following program consists of two modules:

```
/* foo6.c */
void p2(void);

int main()
{
    p2();
    return 0;
```

```
}
```

```
/* bar6.c */
#include <stdio.h>

char main;

void p2()
{
    printf("0x%x\n", main);
}
```

If the program is compiled and executed in Linux, we will see "0x55\n" and the program exits normally. Can you explain the reason why it happens?

The strong symbol associated with the function main in foo6.c overrides the weak symbol associated with the variable main in bar6.c. Thus, the reference to variable main in bar6.o resolves to the value of symbol main, which in this case is the address of the first byte of function main. This byte contains the hex value 0x55, which is the binary encoding of `pushl %ebp`, the first instruction in procedure `main`.

Problem 4

Consider the following C code and its relocatable object file.

```
extern int p3(void);
int x = 1;
int *xp = &x;

void p2(int y) {
}

void p1() {
    p2(*xp + p3());
}
```

```
.text
00000000 <p2>:
       0:  55                          push   %ebp
       1:  89 e5                       mov    %esp, %ebp
       3:  89 ec                       mov    %ebp, %esp
       5:  5d                          pop    %ebp
       6:  c3                          ret
```

```
00000008 <p1>:
       8:  55                      push   %ebp
       9:  89 e5                   mov    %esp, %ebp
       b:  83 ec 08                sub    $0x8, %esp
       e:  83 c4 f4                add    $0xfffffff4, %esp
      11:  e8 fc ff ff ff          call   12 <p1+0xa>
      16:  89 c2                   mov    %eax, %edx
      18:  a1 00 00 00 00          mov    0x0, %eax
      1d:  03 10                   add    (%eax), %edx
      1f:  52                      push   %edx
      20:  e8 fc ff ff ff          call   21 <p1+0x19>
      25:  89 ec                   mov    %ebp, %esp
      27:  5d                      pop    %ebp
      28:  c3                      ret
```

.data

```
00000000 <x>:
   0:  01 00 00 00
00000004 <xp>:
   4:  00 00 00 00
```

A. When the module is relocated, what change will happen in .text section? List their information in relocation entry, including **OFFSET**, **TYPE** and **VALUE.**

Relocation entries for the .text section:

```
1   RELOCATION RECORDS FOR [.text]:
2   OFFSET    TYPE          VALUE
3   00000012  R_386_PC32    p3
4   00000019  R_386_32      xp
5   00000021  R_386_PC32    p2
```

B. When the module is relocated, what change will happen in .data section? List their information in relocation entry, including **OFFSET**, **TYPE** and **VALUE**.

Feel free to use some tools which may be helpful, i.e. objdump.

Relocation entries for .data section:

```
1   RELOCATION RECORDS FOR [.data]:
2   OFFSET    TYPE          VALUE
3   00000004  R_386_32      x
```