Homework 7

Problem 1

Read the following codes and fill the given table.

```
char ch = 'y';
short num[5];
void foo(int x){
        static int i = 7;
        int j = x+1;
}
```

Indicate the location of the following symbols, e.g. .data, .bss, .text, or stack.

Symbol	Location
ch	.data
num	.bss
foo	.text
i	.data
j	Stack

Problem 2

Note: The error check is very important for a robust program.

```
#include <stdio.h>
#inlcude <stdlib.h>
#include <dlfcn.h>
int main() {
    // Your codes here.
    void *handle;
    void (*eat)();
    char *error;
    handle = dlopen("./dog.so", RTLD_LAZY);
    if (!handle) {
        fprintf(stderr, "%s\n", dlerror());
        exit(1);
    }
    eat = dlsym(handle, "eat");
    if ((error = dlerror()) != NULL) {
        fprintf(stderr, "%s\n", dlerror());
        exit(1);
    }
    eat();
    return 0;
}
```

Problem 3

```
/*file: a.c*/
extern int bae(void);
static int x= 1;
int *xp = &x;

void foo2(void);
void foo1(int f){
};

void foo2(void){
foo1(bar() + (int)foop + *xp);
}
```

```
a.o:     file format elf32-i386


Disassembly of section .text:

00000000 <foo1>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   90                      nop
   4:   5d                      pop    %ebp
   5:   c3                      ret

00000006 <foo2>:
   6:   55                      push   %ebp
   7:   89 e5                   mov    %esp,%ebp
   9:   83 ec 08                sub    $0x8,%esp
   c:   e8 fc ff ff ff          call   d <foo2+0x7>
  11:   89 c2                   mov    %eax,%edx
  13:   a1 00 00 00 00          mov    0x0,%eax
  18:   01 c2                   add    %eax,%edx
  1a:   a1 00 00 00 00          mov    0x0,%eax
  1f:   8b 00                   mov    (%eax),%eax
  21:   01 d0                   add    %edx,%eax
  23:   83 ec 0c                sub    $0xc,%esp
  26:   50                      push   %eax
  27:   e8 fc ff ff ff          call   28 <foo2+0x22>
  2c:   83 c4 10                add    $0x10,%esp
  2f:   90                      nop
  30:   c9                      leave
  31:   c3                      ret
Disassembly of section .data:
00000000 <x>:
0:      01 00 00 00
00000004 <xp>:
4:      00 00 00 00
00000008 <foop>:
8:      00 00 00 00
```

If bar is relocated to 0x0804840dand foo2is relocated to 0x080483e1, what will the underlined instruction be changed to after linking?

Foo2 + bias = bar

bias=0x0804840d – (0x080483e1 + 7+ 4)

=0x21

e8 21 00 00 00