1. Book Problem 3.54
2. Book Problem 3.56
3. Book Problem 3.58

4. Consider the following function's assembly code:

```assembly
0040050a <bar>:
  3bf0 00 00 00 00 00 mov $0x0,%ecx
  3bf1 47 03 00 lea 0x3(%rdi),%eax
  3bf2 ff ff 00 cmp $0xffffffffffffffff,%edi
  3bf3 e0 f8 00 cmovle %eax,%edi
  3bf4 d9 00 00 mov %edi,%edx
  3bf5 fa 85 test %edx,%edx
  3bf6 24 7e 14 jle 400530 <bar2+0x17>
  3bf7 00 00 02 85 lea 0x3(%rdx),%eax
  3bf8 ff ff 00 cmp $0xffffffffffffffff,%edx
  3bf9 f7 c1 00 cmovg %edx,%eax
  3bf9 fa 85 test %edx,%edx
  3bf9 16 7e 14 jle 400530 <bar2+0x2b>
  3bfa e1 85 fa 00 lea 0x3(%rdx),%eax
  3bfa fa ff 00 cmp $0xffffffffffffffff,%edx
  3bff c1 fa 02 cmovg %edx,%eax
  3bfc fa 85 sar %edx,%edx
  3bfc 31 7e 14 jle 400530 <bar2+0x2b>
  3bfb 00 00 02 85 lea 0x3(%rdx),%eax
  3bfc 24 7e 14 jle 400530 <bar2+0x2b>
  3bf7 f0 00 00 00 mov %ecx,%eax
  3bf8 c3 00 00 retq
```

Fill in the corresponding C code:

```c
int bar(int x)
{
    int y = 0;
    int z = _________________;
    for(   ;  _______  ;  ___________ )
    {
        z = _________________;
    }
    return ________________;
}
```
5. Given the following assembly code:

```
xorl %eax,%eax
leal 16(%ecx),%ebx
.L59:
  leal (%eax,%eax,4),%edx
  movl (%ecx),%eax
  addl $4,%ecx
  leal (%eax,%edx,2),%eax
  cmpl %ebx,%ecx
  jle .L59
```

Put comments on each assembly statement as it pertains to the C code then give equivalent C code. Here is some additional information that will help:

- **Register assignments:**
  - %ecx is z
  - %eax is zi
  - %ebx is zend

- ** Computations:**
  - $10 * zi + *z is implemented as $*z + 2*(zi+4*zi)$
  - Z++ increments by 4

6. Using the template below (allowing a maximum of 40 bytes, indicate the allocation of data for struct my_struct. Mark off and label the areas for each individual element (arrays may be labeled as a single element) where each cell in the template is 1 byte. Shade the boxes used for padding i.e. the ones that are allocated, but not used; and be sure to clearly indicate the end of the structure. Use the letter of the variable to designate the space filled for each byte.

```
struct my_struct
{  
    short b;
    int x;
    short s;
    long z;
    char c[5]
    long long a;
    char q;
}
```
7. Below is the C code and assembly code for a simple function. Draw a detailed stack diagram for this function, starting with a function that calls this function and continuing for 2 recursive calls of this function i.e. at least two stack frames that belong to this function. Be sure to label everything you can so your solution is understandable.

```c
int doSomething(int a, int b, int c)
{
    int d;
    if (a == 0)
        return 1;
    d = a/2;
    c = doSomething(d, a, c);
    return c;
}
```

```assembly
000000af <doSomething>:
    push %ebp
    mov %esp, %ebp
    sub $0x24, %esp
    mov 0x8(%ebp), %ecx
    mov $0x1, %eax
    test %eax, %eax
    je de <doSomething+0x2f>
    mov %ecx, %edx
    shr $0x1f, %edx
    lea (%ecx, %edx, 1), %edx
    sar %edx
    mov 0x10(%ebp), %eax
    mov %eax, 0x8(%esp)
    mov %ecx, 0x4(%esp)
    mov %edx, (%esp)
    call da <doSomething+0x2b>
    leave
    ret
```

8. Give the IA32 instruction format for each of the following assembly statements.
   A. push %ebp
   B. sub $0x24, %esp
   C. add $0xffffffff8, %esp
   D. lea 0xffffffff8(%ebp), %ebx
   E. push $0x804857b
   F. test %eax, %eax
   G. pop %ebp
   H. ret