#1. a) Write the SPARC assembly instructions to define the following global variables in the data segment:

```assembly
char ModestMouse[] = "Ocean Breathes Salty";
float boat = -4.20;
```

#2. What is the value (in hex) of %o1 after each set of instructions:

a) 
```assembly
set 0x CA2475FE, %o1
set 0x D4A9D4A9, %o2
btog %o2, %o1
```
Value in %o1 at this point is 0x________________________________________

b) 
```assembly
set 0x CA2475FE, %o1
sra %o1, 8, %o1
```
Value in %o1 at this point is 0x________________________________________

c) 
```assembly
set 0x CA2475FE, %o1
set 0x D4A9D4A9, %o2
or %o1, %o2, %o1
```
Value in %o1 at this point is 0x________________________________________

#3. List two reasons why you would need to allocate local variable space on the stack as opposed to mapping a local variable to local registers in the SPARC architecture.
#4. Write the equivalent **unoptimized** SPARC assembly language instructions to perform the following C code fragment. **Use the loop construct specified in class/Notes.**

```c
for ( i = 4920; i > 420; --i ) {
    x = i - x;
}
```

```sparc
SPARC assembly
/* i is mapped to %11 */
/* x is mapped to %13 */
```

#5. Write the equivalent **unoptimized** SPARC assembly language instructions to perform the following C code fragment.

```c
x = x % 5555;
```

```sparc
SPARC assembly
/* x is mapped to %13 */
```

Now optimize your answer to eliminate any delay slots:

**Optimized version of above SPARC assembly**