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

```asm
char dmr[] = "Father of C!";
int code = 42;
```

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

a)  
```
set 0x9DECAFE9, %o1
sra %o1, 12, %o1
```
Value in %o1 at this point is 0x________________________

b)  
```
set 0x9DECAFE9, %o1
set 0x1357ABCD, %o2
xor %o1, %o2, %o1
```
Value in %o1 at this point is 0x________________________

c)  
```
set 0x9DECAFE9, %o1
set 0x1357ABCD, %o2
and %o1, %o2, %o1
```
Value in %o1 at this point is 0x________________________

#3. Assume you run gdb on pa1.
State how to set a breakpoint at the entry point in displaySquare():

Assume you correctly set this breakpoint and performed a run with correct command line arguments. State how to print the value of the 2nd argument passed to displaySquare() in gdb:

(over)
#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 ( x = 23; x <= 420; ++x )
{
    a = x + 42;
}
```

---

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

```c
x = x / 5678;
```

```sparc
SPARC assembly
/* x is mapped to %10 */
```

---

5b. Now optimize your answer from #5a to eliminate any delay slots:

**Optimized version of above SPARC assembly**