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

```
cchar BadReligion[] = "LA Is Burning";
double radio = 94.9;
```

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

a)  
```
sel 0xFEEDBABE, %o1
sra %o1, 8, %o1
```
Value in %o1 at this point is **0x**__________________________________________

b)  
```
sel 0xFEEDBABE, %o1
set 0xC3C3C3C3, %o2
and %o1, %o2, %o1
```
Value in %o1 at this point is **0x**__________________________________________

c)  
```
sel 0xFEEDBABE, %o1
set 0xC3C3C3C3, %o2
xor %o1, %o2, %o1
```
Value in %o1 at this point is **0x**__________________________________________

#3. List two instructions that are not eligible to be used to fill a delay slot 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 = 9021; i >= 17; --i ) {
    x = i + 420;
}
```

```sparc
/* i is mapped to %l2 */
/* x is mapped to %l4 */
```

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

```c
x = x / 5555;
```

```sparc
/* x is mapped to %l4 */
```

Now optimize your answer to eliminate any **delay slots:**

**Optimized version of above SPARC assembly**