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

```
char Death_Cab[] = "April 29";
short stack;
```

#2. What is the value (in hex) of \%01 after each set of instructions:

a) set 0xDEADBEEF, \%01
    sra \%01, 16, \%01

Value in \%01 at this point is 0x______________________________

b) set 0xDEADBEEF, \%01
    set 0x4D3C2B1A, \%02
    xor \%01, \%02, \%01

Value in \%01 at this point is 0x______________________________

c) set 0xDEADBEEF, \%01
    set 0x4D3C2B1A, \%02
    and \%01, \%02, \%01

Value in \%01 at this point is 0x______________________________

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

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 displayX() 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**

```c
for ( a = 9803; a > 181; --a )
{
    x = a - 345;
}
```

**SPARC assembly**

```
/* x is mapped to %l2 */
/* a is mapped to %l4 */
```

---

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

**C**

```c
x = x % 7007;
```

**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**