Signature ____________________

Login name __________________

Quiz 3
CSE 131
Winter 2013

Name ________________________

Student ID ___________________

1. Show the memory layout of the following C struct definition taking into consideration the **SPARC** data type memory alignment restrictions discussed in class. Fill bytes in memory with the appropriate struct member/field name. For example, if member/field name `p` takes 4 bytes, you will have 4 `p`'s in the appropriate memory locations. If the member/field is an array, use the name followed by the index number. For example, some number of `p[0]`s, `p[1]`s, `p[2]`s, etc. Place an `X` in any bytes of padding. Structs and unions are padded so the total size is evenly divisible by the most strict alignment requirement of its members.

```
struct foo {
   int     a;
   char    b;
   short   c[2];
   float   d;
   double *e;
   double  f;
   int     g;
};

struct foo fubar;
```

lower memory

fubar:

higher memory

What is the `offsetof( struct foo, a )`? ________

What is the `offsetof( struct foo, f )`? ________

What is the `sizeof( struct foo )`? ________

If `struct foo` had been defined as `union foo` instead, what would be the `sizeof( union foo )`? ________

2. Given the C array declaration

C

```
double a[3];
```

Mark with an **A** the memory location(s) where we would find `a[1]`

(each box represents a byte in memory)

a:

low memory   high memory

If `a[0]` is allocated at memory location 4000 (decimal), what value does `&a[2]` evaluate to? ___________

3. Given the following Reduced-C definitions (similar to C++)

```
structdef S1 { int a; };
structdef S2 { int a; };

void foo1( int a ) { }
void foo2( S1 &a ) { }

typedef int T1;
typedef T1 T2;
typedef S1 T3;
typedef T3 T4;

S1 a;
S2 b;
T1 c;
T2 d;
T3 e;
T4 f;
float g;
```

indicate whether each of the following statements will cause a compiler error or not.

A) Error
B) No Error

```
  a = e; _____                 foo1( d ); _____
  a = f; _____                 foo1( g ); _____
  a = b; _____                 foo2( a ); _____
  c = d; _____                 foo2( f ); _____
  c = g; _____                 foo2( e ); _____
                               foo2( b ); _____
```

4. Using Reduced-C syntax, define a pointer to an array of 5 floats named `foo` such that

```
float x = 4.2;

(*foo)[4] = x;
x = (*foo)[4];
```

are valid expressions. This will take two lines of Reduced-C code.

5. In the following example

```
typedef int INT;
INT x;
```

What type of STO do you need to create for `INT`? ________________________

What type of STO do you need to create for `x`? ________________________