Signature _____ **Quiz 3** **Name** _____

CSE 131

Login name _____ **Winter 2012** **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 {
   char    a;
   double  b;
   short   c[3];
   double *d;
   short   e;
   char    f;
};

struct foo fubar;
```

What is the `offsetof( struct foo, b )`? _____

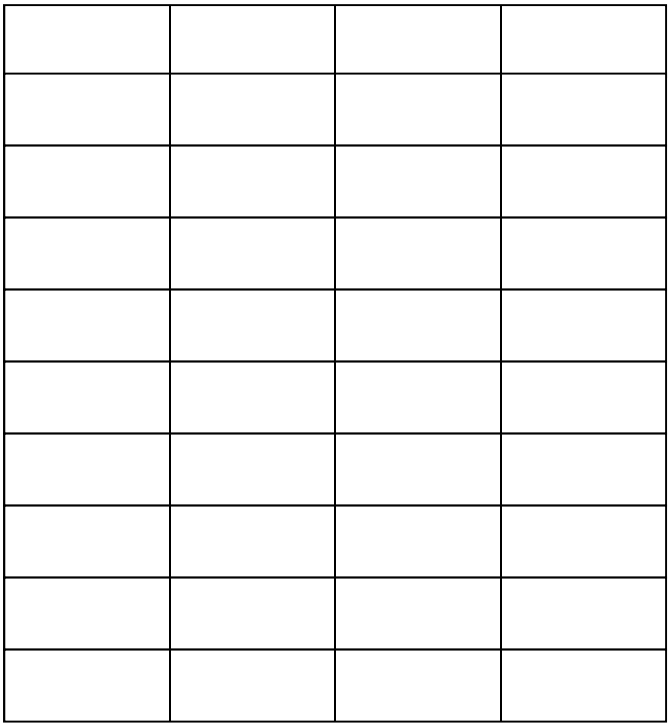
What is the `offsetof( struct foo, c[1] )`? _____

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

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

lower memory

fubar:



higher memory

2. Given the C array declaration

C

```
double *a[7];
```

Mark with an **A** the memory location(s) where we would find `a[5]`

(each box represents a byte in memory)

a:

low memory high memory

If `a[0]` is allocated at memory location 6000, what value does `&a[4]` evaluate to? _____

3. The types in Reduced-C variable definitions are often unnecessary in the sense that it may be possible to infer variables' types and detect type errors simply from their use. For the following program fragment, find a set of types that makes it legal, and write a Reduced-C definition for each variable. If there is more than one possible type, choose only one. If there is none, write "NONE".

```
function : bool foo( int &x ) {...}

if( c = (d == foo( a )) )
    b = &a;

_____________________ a ;

_____________________ b ;

_____________________ c ;

_____________________ d ;
```

```
a = 42.420;

if( b != c )
    b = (d ^ c) / a;

_____________________ a ;

_____________________ b ;

_____________________ c ;

_____________________ d ;
```

4. Using the Rt-Lt Rule, give the C/C++ variable definition for a variable named cafe that is a pointer to a function that takes a pointer to short as its only argument and returns a pointer to an array of 4 elements where each element is a pointer to a pointer to an int.

5. Define an array of array of ints named `bar` in Reduced-C such that

```
bar[8][4]
```

is the last element in this data structure. You will need two lines of Reduced-C code to do this.

6. Given the following Reduced-C code below, fill in the blanks of the Check 5 compile error that should be reported according to this quarter's Project I spec. Use the letters associated with the words in the box below.

1. 5-hour Energy	8. Sleep	15. equivalent
2. modifiable	9. addressable	16. assignable
3. I2	10. I1	17. int
4. F1	11. F2	18. float
5. argument	12. value	19. reference
6. x	13. y	20. z
7. a	14. b	

```
typedef float F1;
typedef F1 F2;
typedef int I1;
typedef I1 I2;

F1 x;
I2 y;
F2 z;

I2 foo( F1 a, I1 & b ) { return b; }

x = foo( y, z ); // Compile error reported here. Assume this stmt is inside a function.
```

____ of type ____ not ____ to ____ parameter ____ of type ____ .