#1. Given

```c
int a;
void fubar( int b )
{
    int *c = (int *) malloc( 4 );
    ...
}
```

When this function is called, identify which area of the C Runtime Environment each of the following will be allocated and its scope or visibility.

<table>
<thead>
<tr>
<th>Area of Runtime Env.</th>
<th>Scope/Visibility (Global/File/Function)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td></td>
</tr>
<tr>
<td>fubar</td>
<td></td>
</tr>
</tbody>
</table>

Where c is pointing __________________

If the function above is called 6 times, how many times will c be initialized?  _________

Using the Rt-Lt Rule, define a variable named quiz4 that is a pointer to an array of 42 elements where each element is a pointer to an array of 17 elements where each element is a struct bar.

#2. a) Convert 108.625₁₀ to binary fixed-point and single precision IEEE floating-point representation (expressed in hexadecimal).

binary fixed-point _____________________________ x 2⁰

IEEE floating-point _____________________________ (hexadecimal)

b) Convert 0xC21D8000 (single precision IEEE floating-point representation) to fixed-point decimal.

fixed-point decimal _____________________________ (decimal / no exponential notation)
#3. What is the output of the following program? (Hint: Draw stack frames!)

```c
void swap1( int a, int b )
{
    int tmp;
    tmp = a;
    a = b;
    b = tmp;
}

void swap2( int *a, int b )
{
    int tmp;
    tmp = *a;
    *a = b;
    b = tmp;
}

void swap3( int *a, int *b )
{
    int tmp;
    tmp = *a;
    *a = *b;
    *b = tmp;
}

int main()
{
    int a = 6;
    int b = 2;
    swap1( a, b );
    printf( "%d\n", a );
    printf( "%d\n", b );
    a = 12;
    b = 3;
    swap2( &a, b );
    printf( "%d\n", a );
    printf( "%d\n", b );
    a = 5;
    b = 10;
    swap3( &a, &b);
    printf( "%d\n", a );
    printf( "%d\n", b );
    return 0;
}
```

#4. What gets printed with the function call `mystery( 6 );`? (Hint: Draw stack frames!)

```c
int mystery( int param )
{
    int local = 2;

    if ( local <= param )
    {
        local = local + param;
        printf( "%d\n", local ); /* Output the value of local followed by a newline */
        param = mystery( local - 4 ) + local;
        printf( "%d\n", param ); /* Output the value of param followed by a newline */
    }
    else
    {
        printf( "Whoa\n" );
    }

    return local;
}
```