#1. In the SPARC architecture, to access local variables allocated on the stack you would use

a ___________________ (positive or negative) offset relative to register %_____.

The _________________ SPARC instruction saves the current value of the program counter into %o7.

The _________________ SPARC instruction adds ____ to the value in register %_____ and sets the result into register %_____.

The _________________ SPARC instruction causes the current register set window to slide down 16 registers within the Register Bank. When this instruction completes execution, the old register set

registers %_____ — %_____ are now mapped to the new register set registers %______ — %______.

Using the Rt-Lt Rule, define a variable named foo that is an array of 31 elements where each element is a pointer to a function that takes a pointer to a struct foo and returns a pointer to double.

#2. a) Convert 133.75\textsubscript{10} to binary fixed-point and single precision IEEE floating-point representation (expressed in hexadecimal).

binary fixed-point _________________________________ x 2^0

IEEE floating-point _________________________________ (hexadecimal)

b) Convert 0xC2ED4000 (single precision IEEE floating-point representation) to fixed-point decimal.

fixed-point decimal _________________________________ (decimal / no exponential notation)

(over)
#3. What is the output of the following program? (Hint: Draw stack frames!)

```c
int main()
{
    int a = 4;
    int b = 9;
    swap1( &a, b );
    printf( "%d\n", a );
    printf( "%d\n", b );
    a = 7;
    b = 13;
    swap2( a, &b );
    printf( "%d\n", a );
    printf( "%d\n", b );
    a = 2;
    b = 8;
    swap3( a, b );
    printf( "%d\n", a );
    printf( "%d\n", b );
    return 0;
}
```

#4. What gets printed with the function call `mystery( 2 );`? (Hint: Draw stack frames!)

```c
int mystery( int param )
{
    int local = 8;

    if ( local >= param )
    {
        local = local + param;
        printf( "%d\n", local );  /* Output the value of local followed by a newline */
        param = mystery( local - 5 ) + local;
        printf( "%d\n", param );  /* Output the value of param followed by a newline */
    }
    else
    {
        printf( "Stop\n" );
    }

    return local;
}
```