1. Short Answer (15 points)

   (a) If a 4-byte word contains the integer value 0x1234, what value is stored at the highest numerical address in big endian format? (5 points)

   (b) Name two things that the `leal` instruction are commonly used for in x86 assembly. (5 points)

   (c) Name a situation in which the compiler might favor a sequence of branches to a jump table for implementing a C `switch` statement. (5 points)

   (d) Give a plain-English description of the data structure in the C declaration "`int (*messy[4])[5]`". (5 points)

2. Given the structure definition:

```
struct a {
    int i;
    short s;
    char c;
    int *p;
};
```

   What offset would the pointer p be at in this structure, assuming the compiler inserts the minimal padding needed to ensure natural alignment of each structure field? Justify your answer. (10 points)

3. Consider an 8-bit IEEE-style floating point number with 5 bits of exponent and 2 bits of fraction.

   (a) What bias is used to represent the exponent? (2 points)

   (b) What are the largest and smallest positive normalized numbers that can be represented in this format? (8 points)

   (c) What are hexadecimal representations of -0, +Infinity, and -NaN in this format? (5 points)

   (d) What is the decimal value of the bit pattern 0xf3 in this foating point format? (5 points)

4. Consider the body of a function with the C prototype "int f(int *a, int n);" and the 64-bit X86 assembly listed below:

```
# Type signature: int f(int *a, int n).
# Arg1 in %rdi, Arg2 in %edi; Return value in %rax
f:
        movl    $0, %eax
        movl    $0, %edx
        jmp     .L2                    # Question 5c
.L3:
        movslq  %edx, %rcx              # mov long to quad, sign extend
        leaq    (%rsi,%rcx,4), %rcx
        addl    (%rcx), %eax
        movl    %eax, (%rcx)
        addl    $1, %edx
.L2:
        cmpl    %edi, %edx              # Question 5b
        jl      .L3                     # jump less than
        rep ret
```

   (a) This function uses 2 general-purpose registers outside of the arguments and return address, namely `%rcx` and `%rdx`. Based on how this function handles them, are they caller-save or callee-save? Justify your answer. (5 points)

   (b) What instruction sets the condition codes used by the jump instruction labeled "Question 5b"? (5 points)

   (c) Write C-like pseudocode (without using goto statements) for the loop starting at the line labeled "Question 5c"? (10 points)