1. Can the zero flag (ZF) and the sign flag (SF) be both 1 at the same time? If yes, give an example of an operation that does this (no need for assembly code, just describe the operation). If not, explain why not.

2. Suppose we have the following code (assuming a, b, and b are unsigned integers):

   ```
   if( a == b && b > c)
       c = c + a + b;
   ```

   Write the corresponding assembly code, assuming:
   a will go in %eax, b in %ebx, and c in %ecx)

3. Fill in the blanks in the following table, assume the instructions are executed sequentially (i.e. instruction 1 executed, then instruction 2, then instruction 3. Assume rax and rbx are holding signed numbers.

<table>
<thead>
<tr>
<th>instruction#</th>
<th>instruction</th>
<th>rax</th>
<th>rbx</th>
<th>CF</th>
<th>ZF</th>
<th>SF</th>
<th>OF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initially</td>
<td></td>
<td>0xFFFF00001</td>
<td>0x00000000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>addq %rbx, %rax</td>
<td>0xFFFF00001</td>
<td>0x00000000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>testq %rbx, %rax</td>
<td>0xFFFF00001</td>
<td>0x00000000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>cmpq %rbx, %rax</td>
<td>0xFFFF00001</td>
<td>0x00000000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
4. Consider the following assembly representation of a function `foo` containing a `for` loop:

```assembly
foo:
    movl %edi, %ebx
    leal 2(%ebx), %edx
    xorl %ecx, %ecx
    cmpl %ebx, %ecx
    jge .L4
.L6:
    leal 5(%ecx, %edx), %edx
    leal 3(%ecx), %eax
    imull %eax, %edx        # this means edx = edx * eax
    incl %ecx              # this means ecx++
    cmpl %ebx, %ecx
    jl .L6
.L4:
    movl %edx, %eax
    ret
```

Fill in the blanks in the following code to provide the functionality of the loop:

```c
int foo(int a){
    int i;
    int result = -------;
    for(----; ---- ; i++ )
    {------------------;
        ------------------ ;
    }
    return result;
}
```