1. Assume the CPU is executing a program and the state of some of its registers is given in the table below. Show how the registers would be updated by the sequence of IA32 instructions also listed below, i.e. fill in the Final Value column. Show your work by listing the intermediate values of the registers.

<table>
<thead>
<tr>
<th>Register</th>
<th>Initial Value</th>
<th>Final Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>%eax</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>%ebx</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>%ecx</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>%edx</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Here are the IA32 instructions:

- `addl $20, %eax`
- `addl %eax, %ebx`
- `subl %ecx, %ebx`
- `addl $3, %ecx`
- `subl %edx, %ecx`
- `addl %edx, %edx`
- `decl %edx`
- `shrl $4, %ebx`
- `andl $0xfffffffffe, %edx` # this is tricky
- `xorl %eax, %eax` # this is tricky
- `orl $0x0, %ecx`
  # think about these next two before answering
- `notl %ebx`
- `addl $1, %ebx`
2. Assume the CPU is executing a function that has local variables x, y, and z allocated on the stack, and that x is allocated at the memory address that is -12 bytes from the address value stored in register %ebp, or -12(%ebp). Assume y is stored at -8(%ebp), and z is at -4(%ebp).

For the assembly code and register values listed below:

(1) Show the values that will be stored in the registers and in memory when execution of these instructions is complete. If the value is unknown, write "?".

(2) Write a C code translation of the assembly code sequence. You may assume that x, y, and z have already been declared as int variables in the C code. You do not need to write the entire function, just the lines of C that might have generated the IA32 instructions. Hint: our solution is 5 lines of C code.

### Memory Address | Final Value
--- | ---
0xff38 | 
0xff3c | 
0xff40 | 
0xff44 | 
0xff48 | 
0xff4c | 
0xff50 | 

### Register | Initial Value | Final Value
--- | --- | ---
%eax | 4 | 
%edx | 7 | 
%ebp | 0xff44 |