Consider the following piece of x86-64 assembly code, where each instruction is assigned an ID:

1. mov -8(%rbp), %r10
2. add %r15, %r10
3. mov %r10, -16(%rbp)
4. mov -24(%rbp), %r11
5. add %r11, %r12
6. mul %r10, %r12
7. add %r11, %r12

Each memory instruction completes in 2 cycles, each arithmetic instruction completes in 1 cycle.

Consider the simple processor without the instruction pipeline. In how many cycles does the computation execute?

Consider now a superscalar processor with the instruction pipeline. Reorder the instructions below to minimize the execution time. In how many cycles does the computation execute?

<table>
<thead>
<tr>
<th>Cycle</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(In each cell, fill in the id of the instruction that starts in the corresponding cycle)