For each expression or set of expressions, give the value and type of the value returned by evaluating the last expression in the set.

<table>
<thead>
<tr>
<th>number</th>
<th>expression</th>
<th>value</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>((lambda (x) (+ x y)) 7)</td>
<td>7</td>
<td>(lambda x y)</td>
</tr>
<tr>
<td>2.</td>
<td>((lambda (x) (let ((y 4)) (+ x y))) 7)</td>
<td>7</td>
<td>(lambda x)</td>
</tr>
<tr>
<td>3.</td>
<td>(lambda (x) (x 4 5) )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>(lambda (a b c) (+ a b))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>(lambda (x y) (lambda (x) (y x)))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>( ((lambda (x y) (lambda (z) (x y z)) ) + 2) 4)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
7. \((\lambda (x)\)
   \((\text{let}\ ((a\ 1))\)
   \((b\ 5))\)
   \((\text{if}\ x\ a\ b))\)
\)
\((>\ 20\ 10))\)

8. \((\text{define}\ x\ +)\)
\((\text{let}\ ((a\ 3))\)
\((\text{list}\ x\ a\ a))\)

9. \((\text{define}\ \text{foo}\ a\ b)\)
\((\text{let}\ ((x\ 6))\)
   \((c\ (+\ a\ 5)))\)
\((+\ b\ x\ c)))\)
\((\lambda (x\ y\ f)\)
\((f\ x\ y))\)
\(1\ 2\ \text{foo})\)

Extra problem (not to worry about now):
10. \((\text{let}\ ((a\ 10))\)
\((b\ 2))\)
\((\text{let}\ ((c\ (+\ a\ b)))\)
\((*\ a\ c))\)

Note: The second let is needed because the value of a variable is not bound until the entire list of variable-value pairs is evaluated. In this example, the value of a or b can't be used in defining c in the first let's list of variables.