### Union-Find Example

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>initial state</strong> find(1) = red</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>union(0,2)</strong></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td><strong>union(1,3)</strong></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td><strong>union(4,5)</strong></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td><strong>union(2,1)</strong></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>find(1) = green</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Solution 1: Quick-Find

- Select a concrete representation
  - array A of N (i.e., |V|) integers
  - A[i] = name of set containing element i
  - Initial state: A[i] = i for all i

- Define concrete operations
  - find(i) is easy: return A[i];
  - runs in constant time

### Solution 1: Implementing Union

```plaintext
union(p, q)
    let firstcolor = A[p];
    let secondcolor = A[q];
    for each element in A
        if (A[i] == firstcolor)
            A[i] = secondcolor
```

- runs in time proportional to N for each union
- If you have $10^6$ connections among $10^9$ nodes...you lose!