Algorithm 1: Composing p_1 and p_2 to p, by a given partial mapping

Input: $p_1 = (L_1, K_1, R_1)$
Input: $p_2 = (L_2, K_2, R_2)$
Input: μ, a partial matching between L_2 and R_1
Output: $p = (L, K, R)$

```
1  p ← empty rule
2  Copy vertices of p1 to p
3  foreach vertex v ∈ p2 do
4  |  if v is not mapped by μ then
5  |  |  Copy v to p
6  |  else
7  |  |  Change membership in L, K and R for vertex μ(v)
8  Copy edges of p1 to p
9  foreach Edge e ∈ p2 do
10 |  if e is not mapped by μ then
11 |  |  Copy e to p
12 |  else
13 |  |  Change membership in L, K and R for edge μ(e)
14 Delete edges and vertices created by p1, but deleted by p2
15 if matching condition not satisfied then abort
16 return p
```