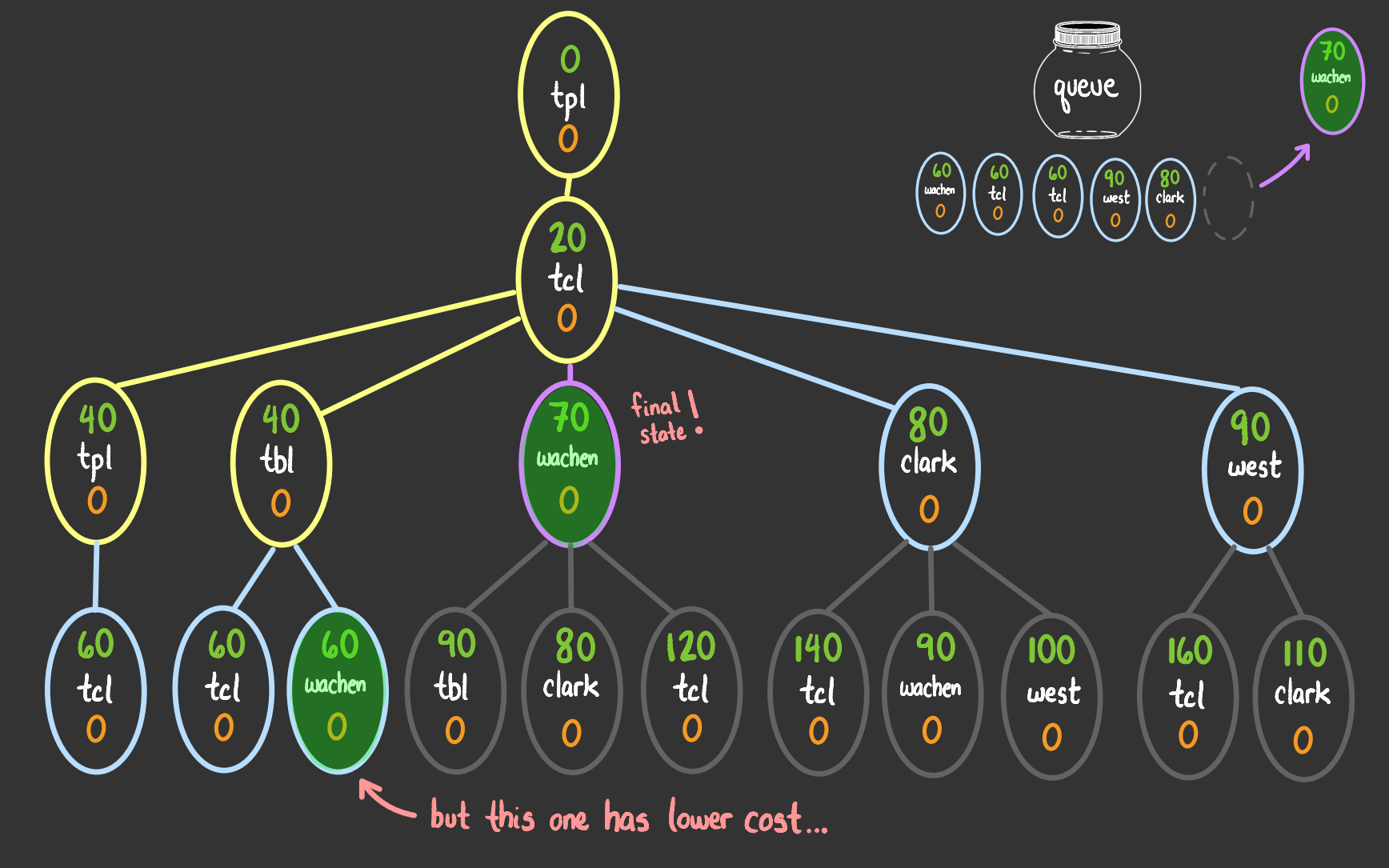
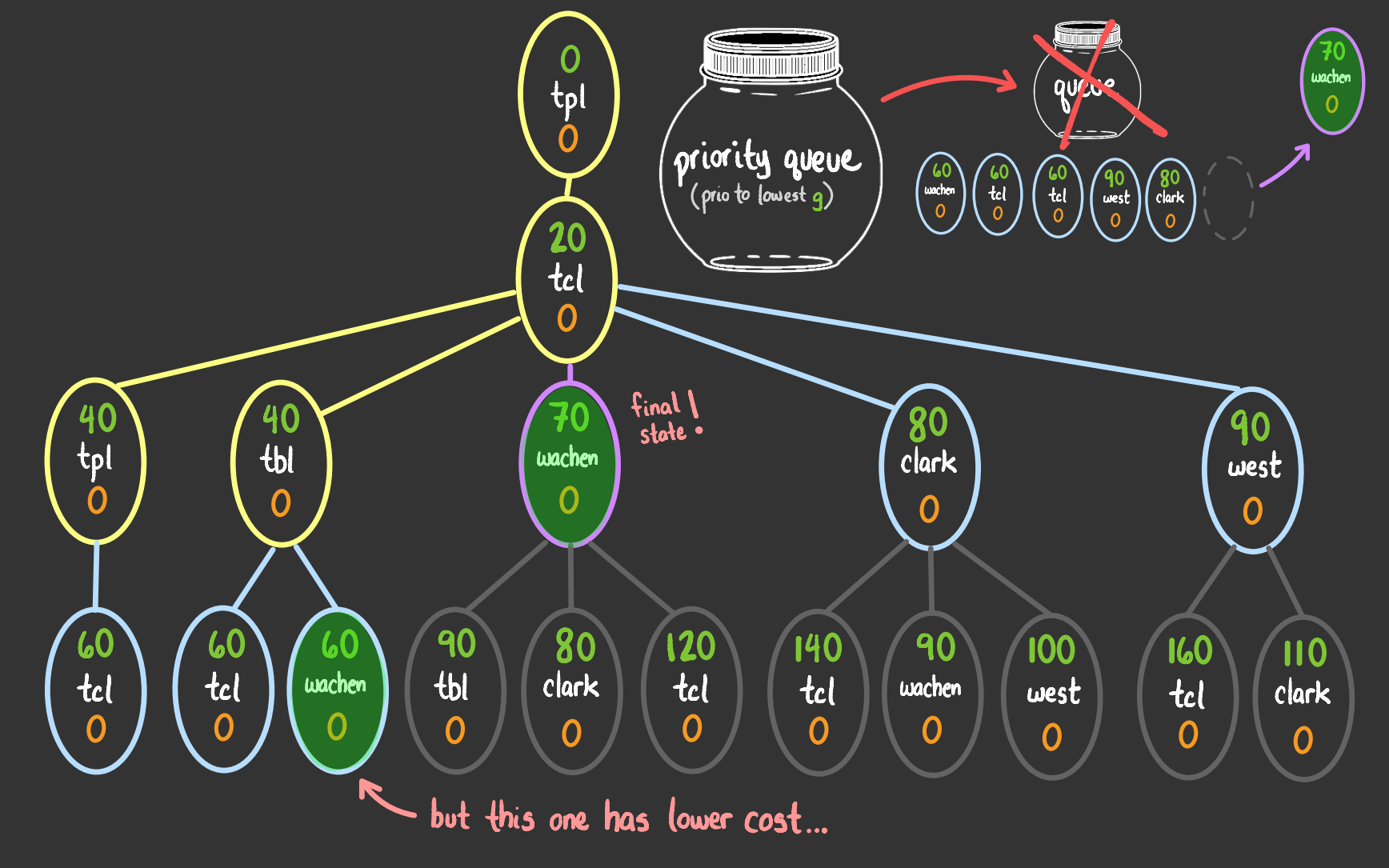
uniform cost search

16 sept 2022

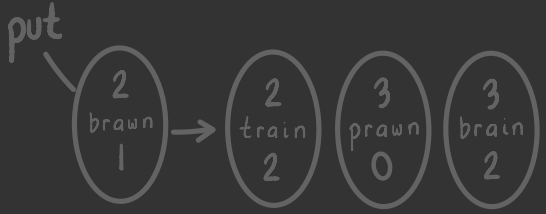
CSCI 373

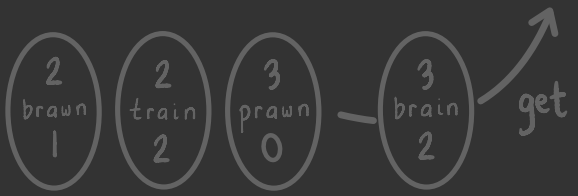

0
tpl
0
queue
70
wachen
0
60
wachen
0
60
tcl
0
60
tcl
0
90
west
0
80
clark
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
final ! state .
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0
but this one has lower cost...


0
tpl
0
priority queue
(prio to lowest g)
queue
60
wachen
0
60
tcl
0
60
tcl
0
90
west
0
80
clark
0
70
wachen
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
final ! state .
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0
but this one has lower cost...

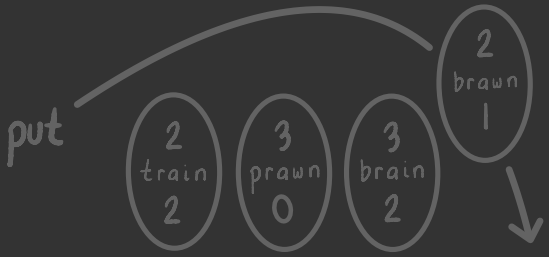
example containers

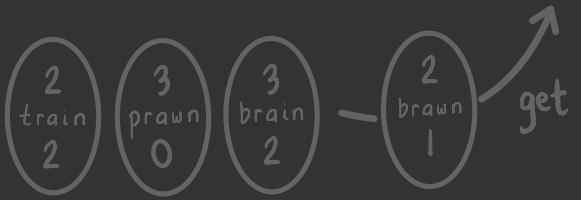
queue



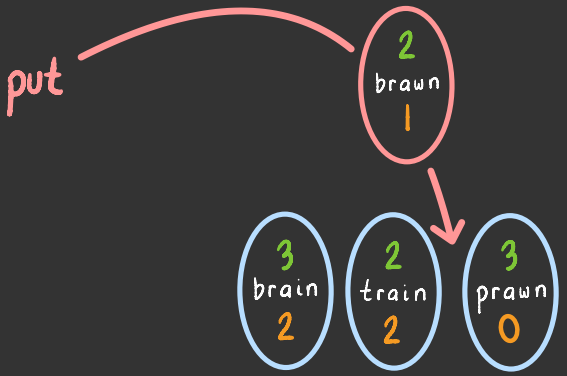


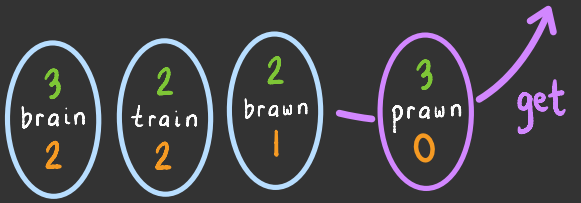
stack





priority queue



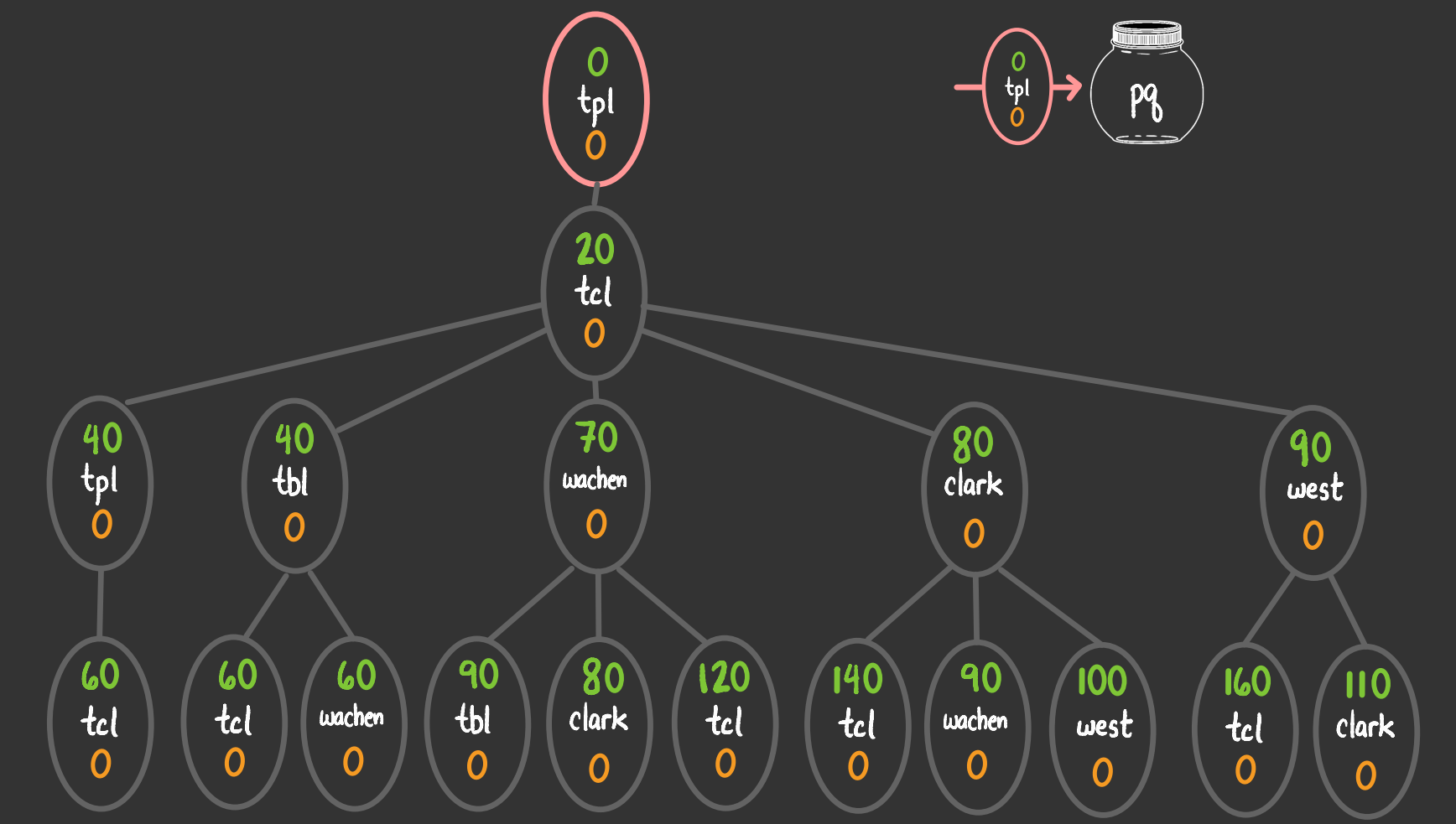


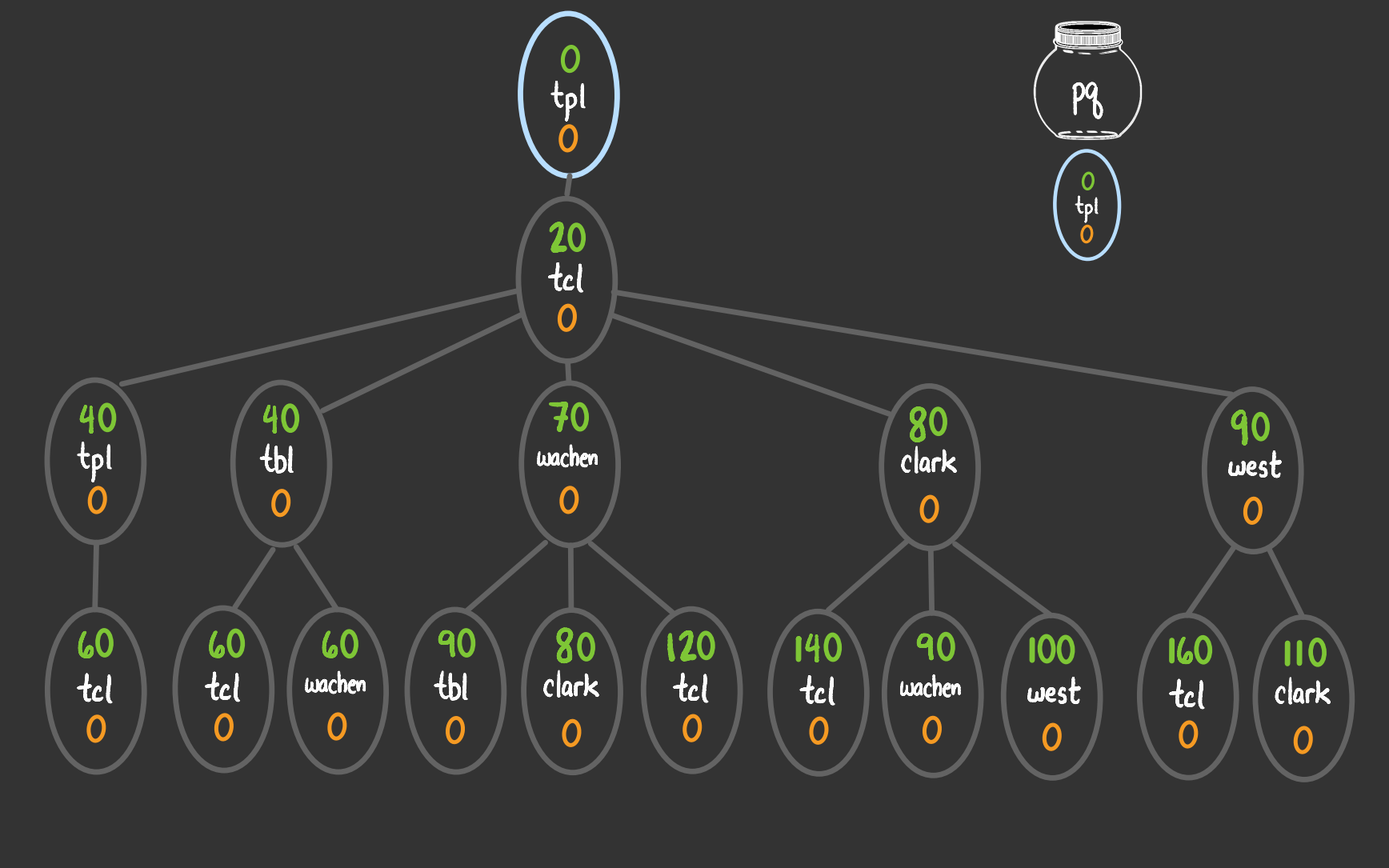
SEARCH$(M = (Q, \Sigma, \Delta, q_0, F, w), H)$:

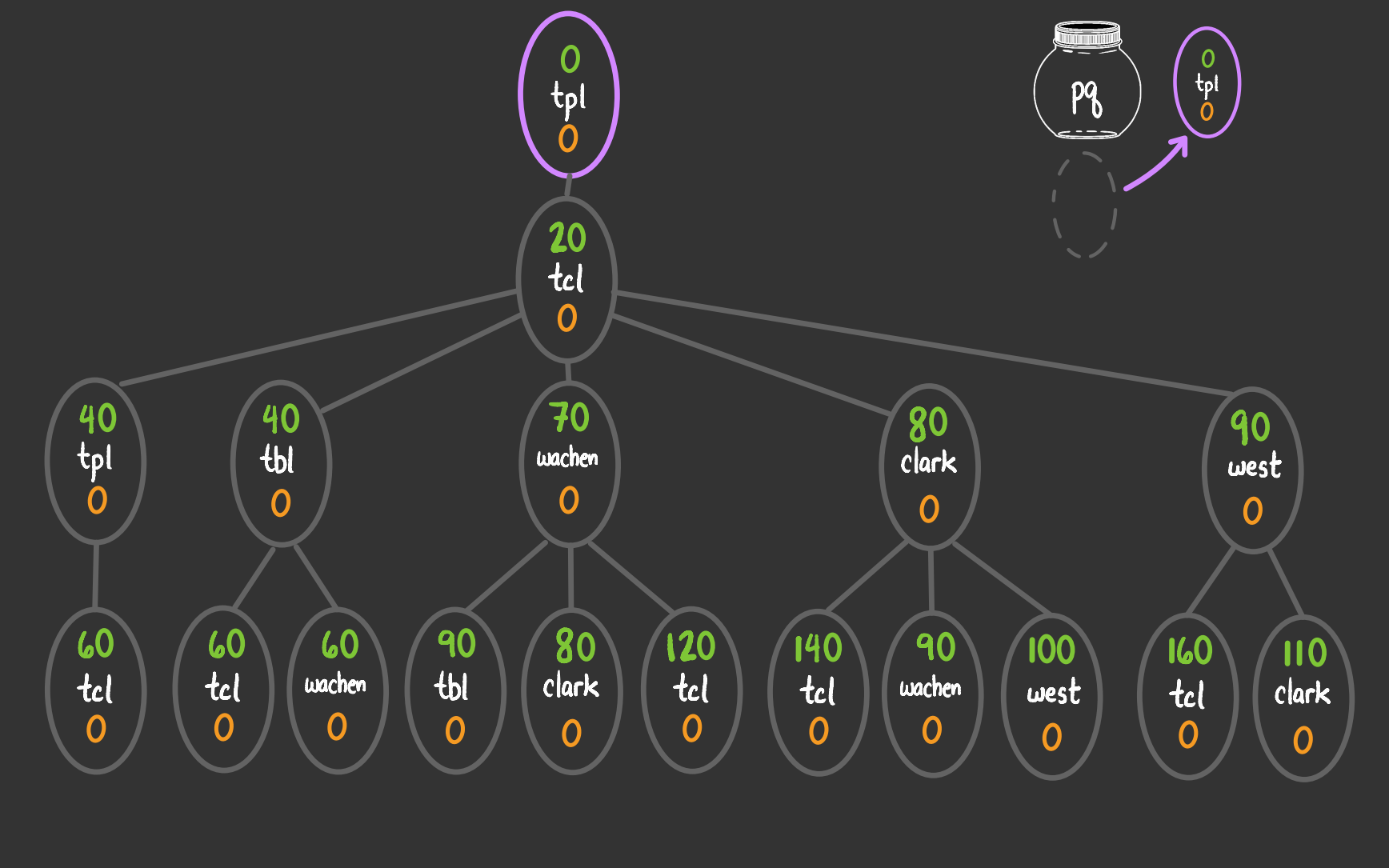
- container = new Container()
- container.put($\langle q_0, 0, H(q_0) \rangle$)
- repeat:
  - if container.empty() then return ∞
  - n = container.get()
  - if $q(n) \in F$ then return $g(n)$
  - let $\text{successors}_{M,H}(n) = \{ \langle q', g(n) + w(q(n), \sigma, q'), H(q') \rangle \mid (q(n), \sigma, q') \in \Delta \}$
  - for $n' \in \text{successors}_{M,H}(n)$:
    container.put(n')

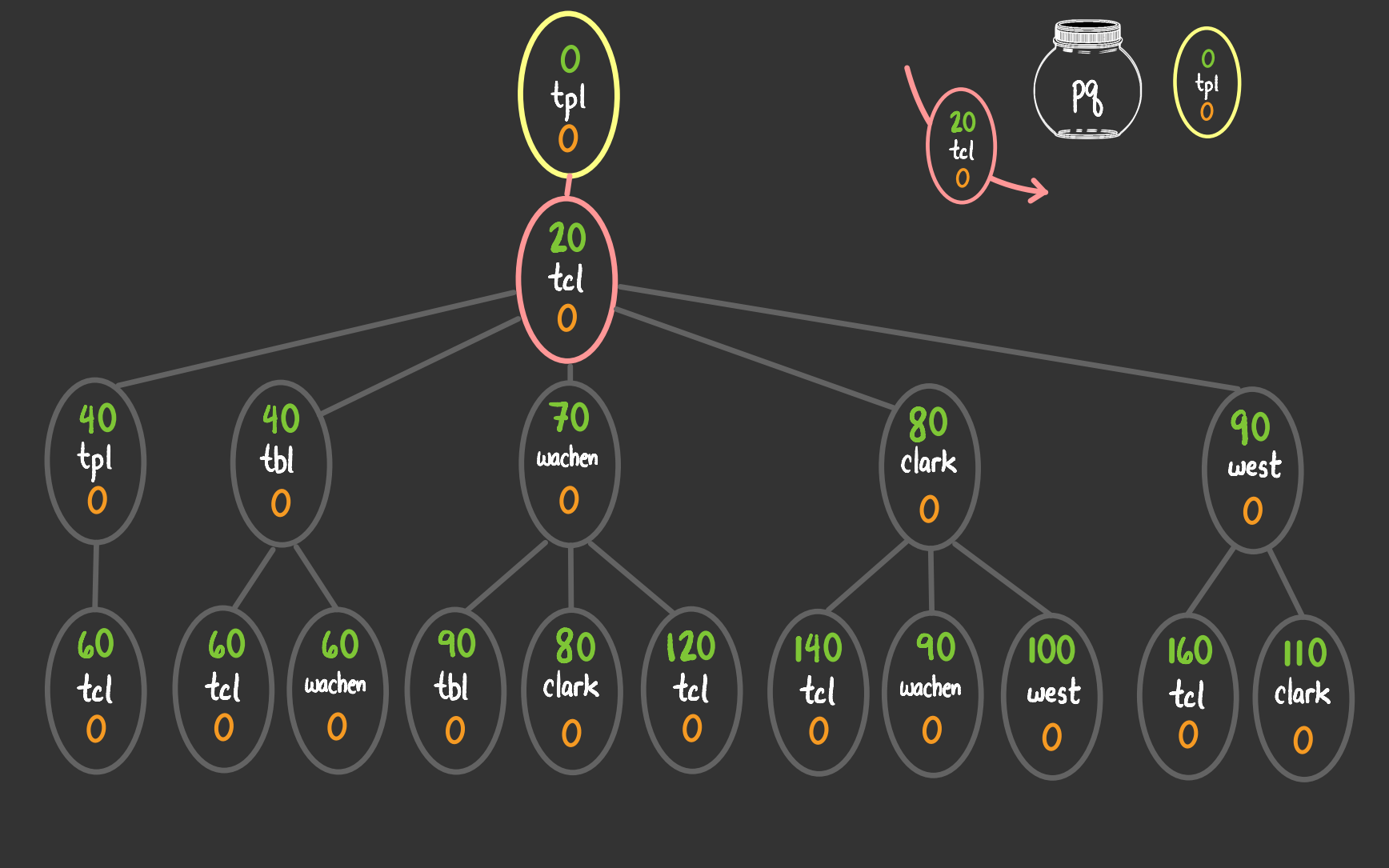
"visit" a node

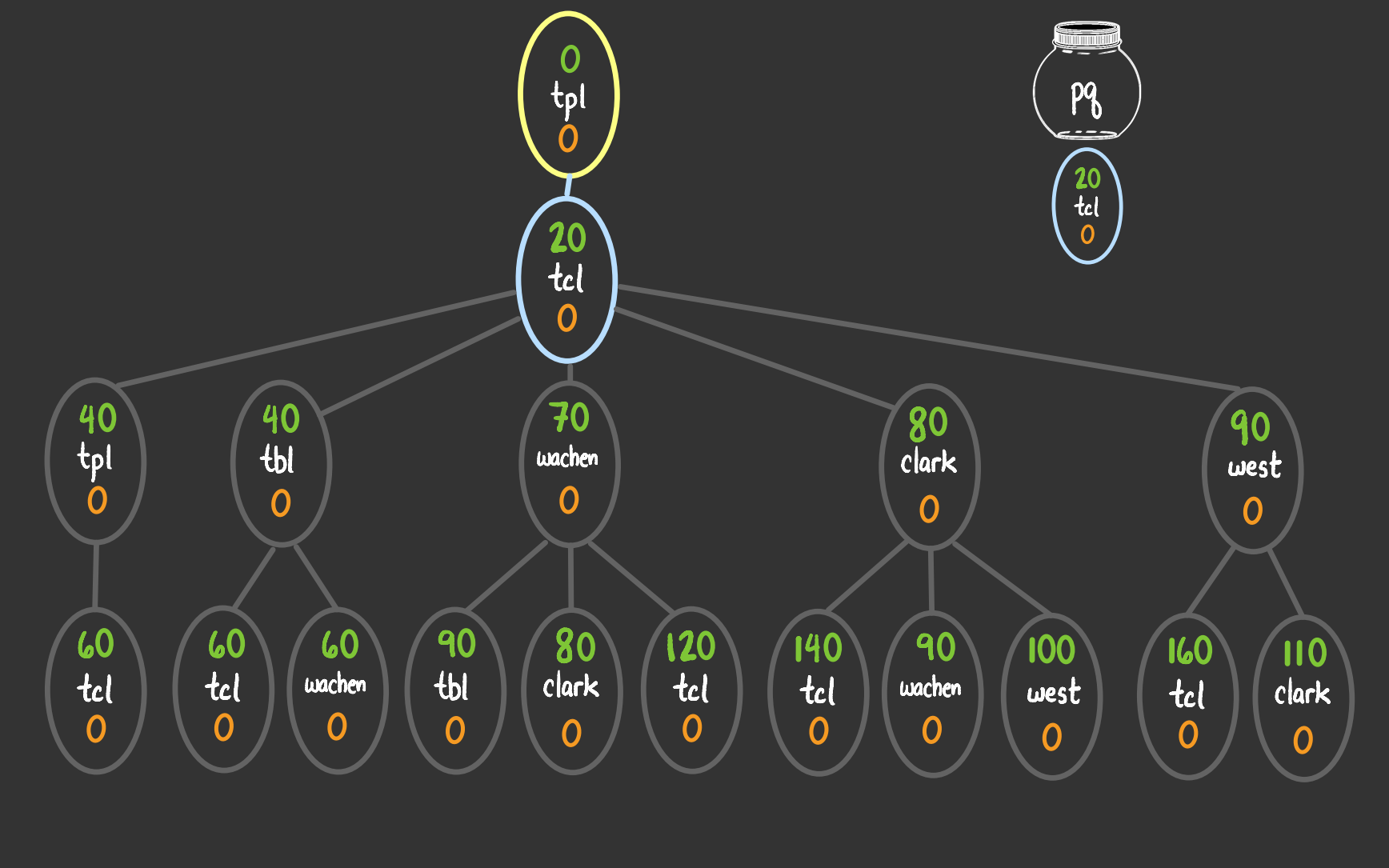
feel free to play along...

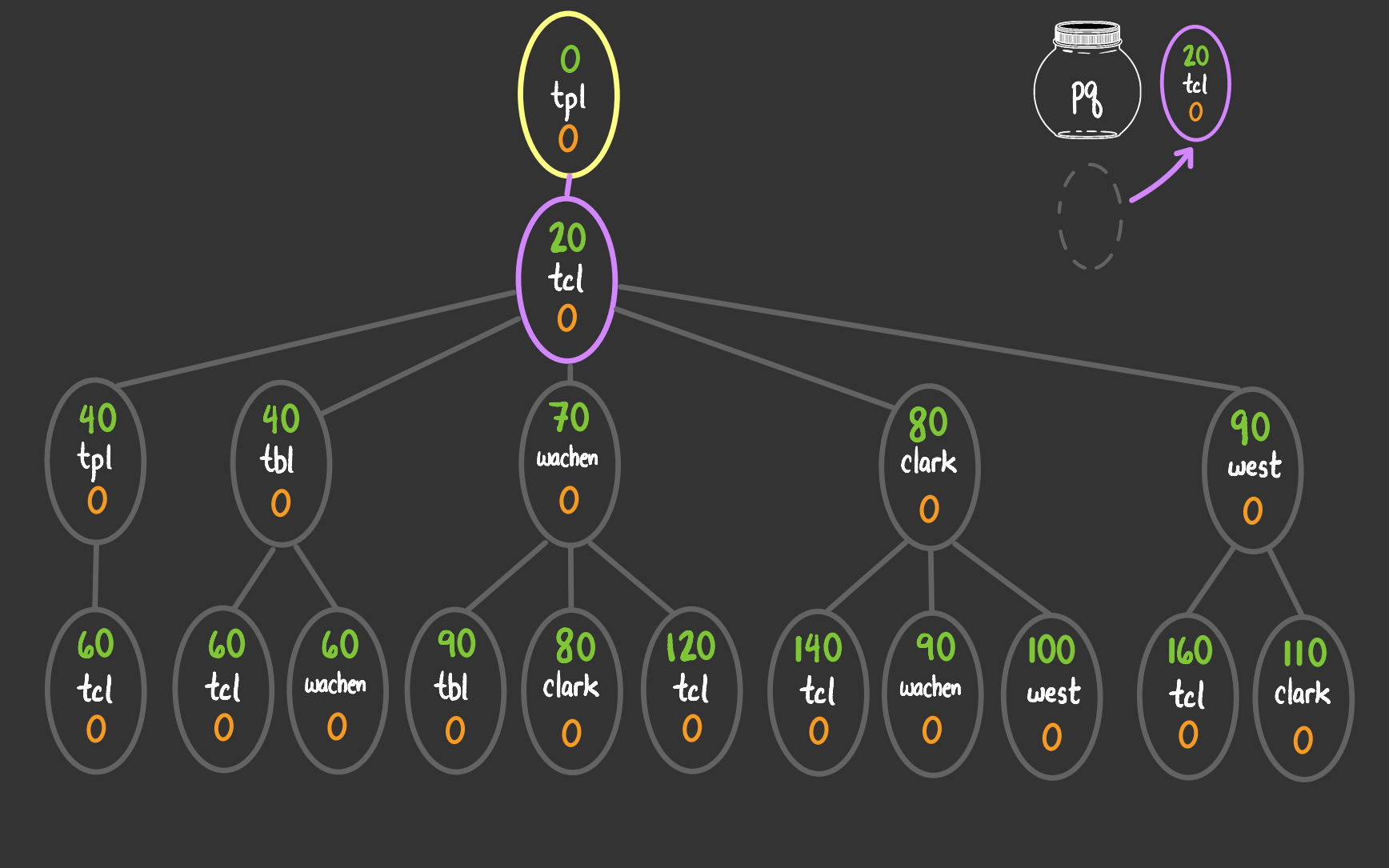

0
tpl
0
0
tpl
0
pq
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

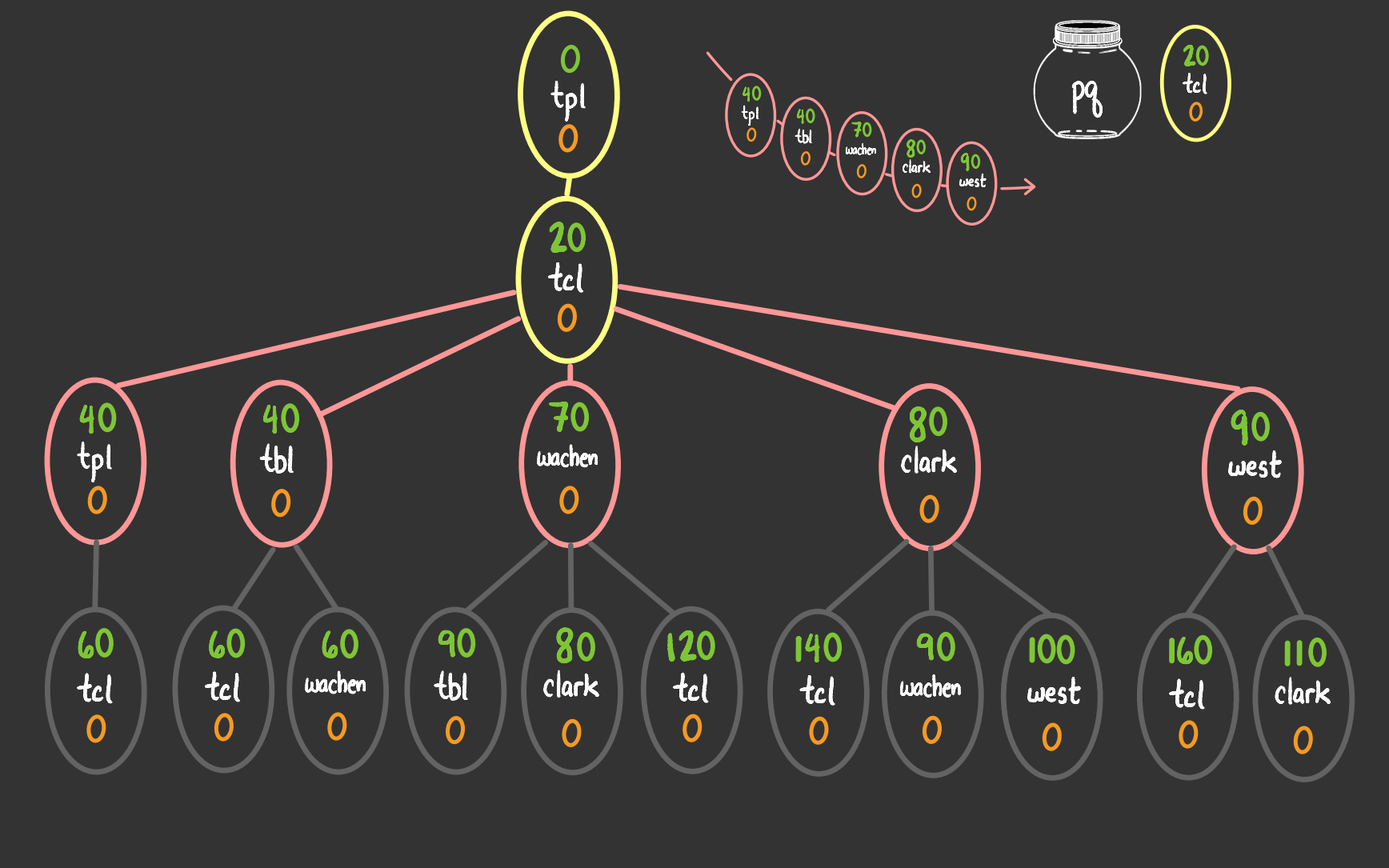

0
tpl
0
pq
0
tpl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

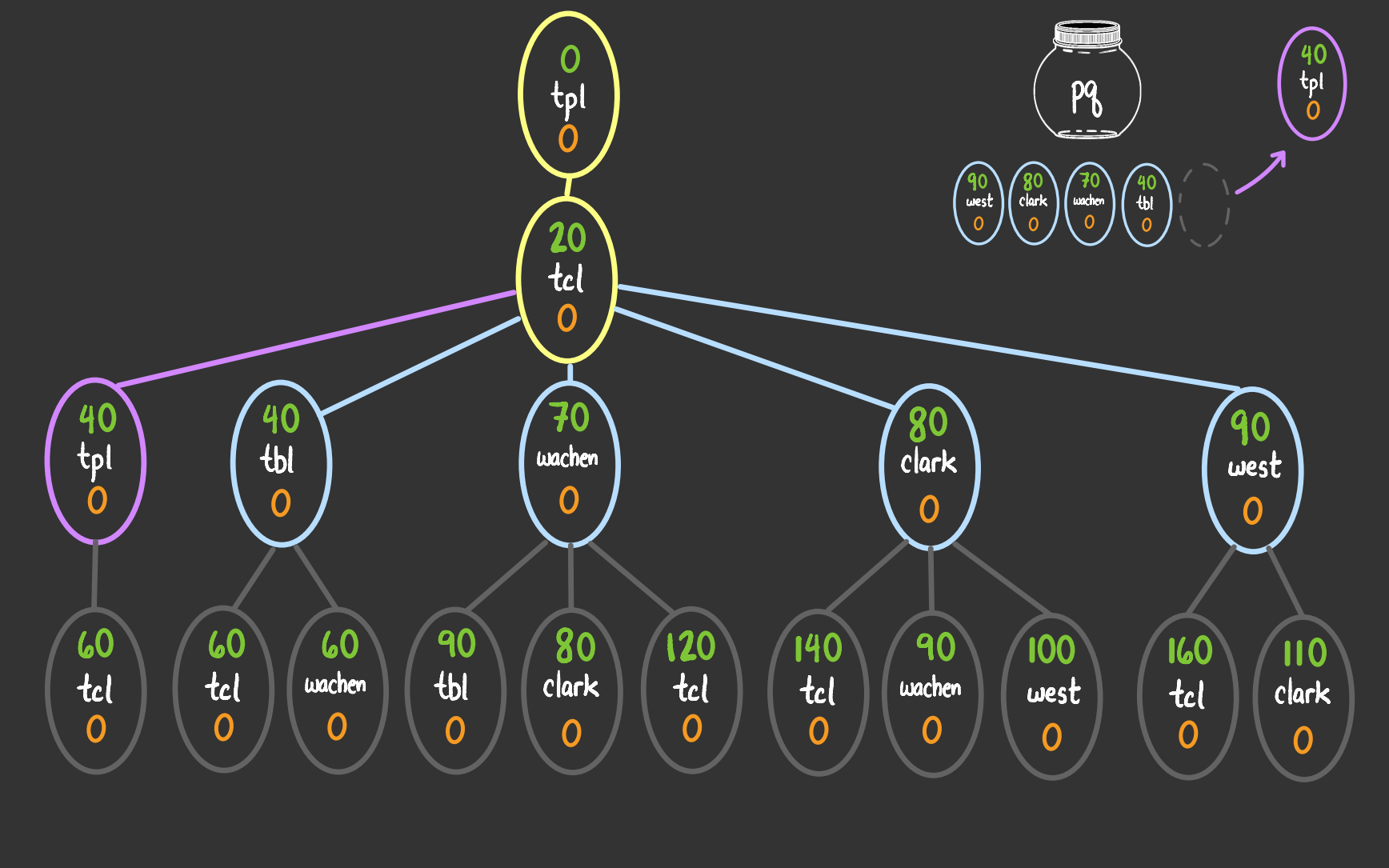

0
tpl
0
pq
0
tpl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

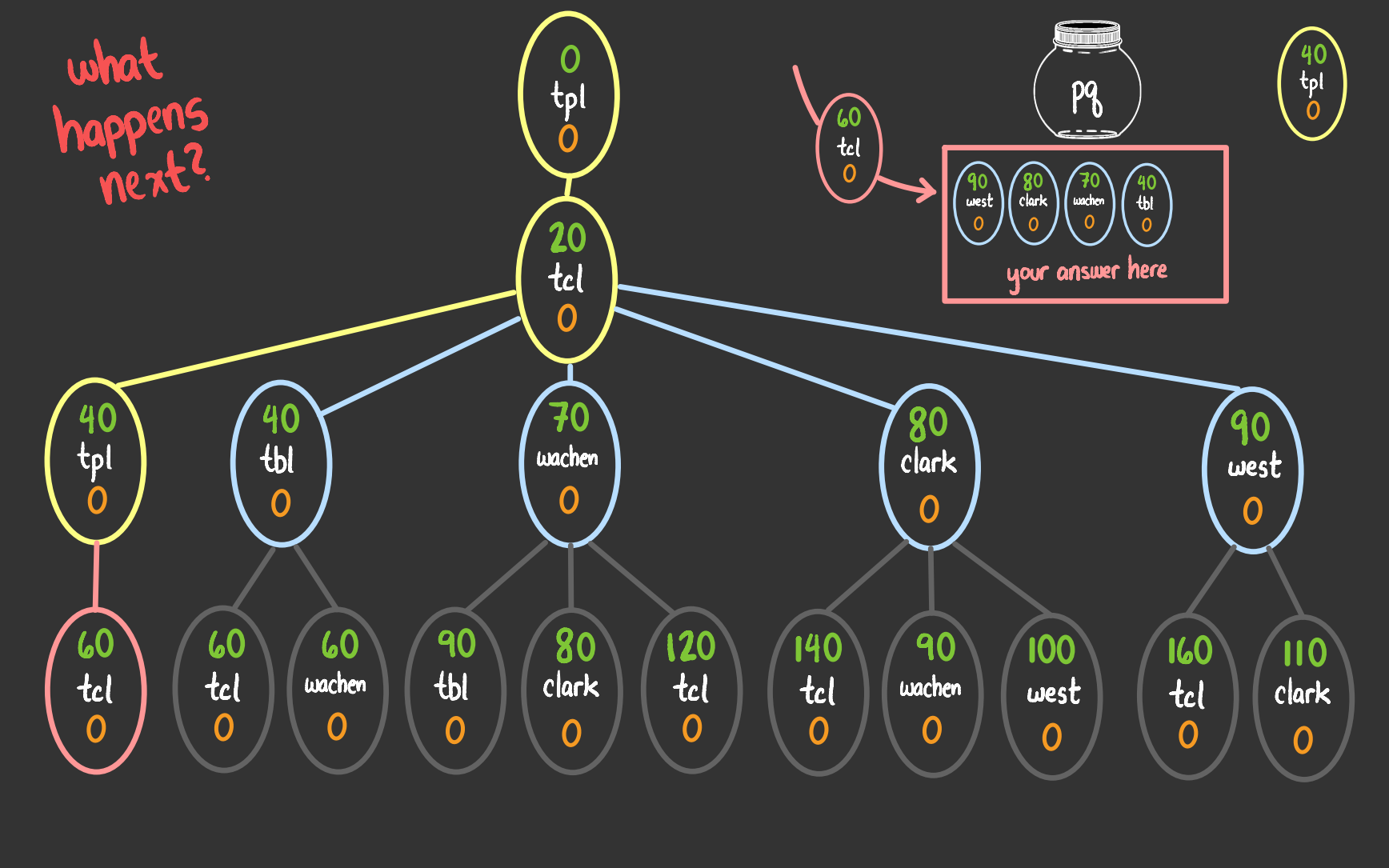

0
tpl
0
20
tcl
0
pq
20
tcl
0
0
tpl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

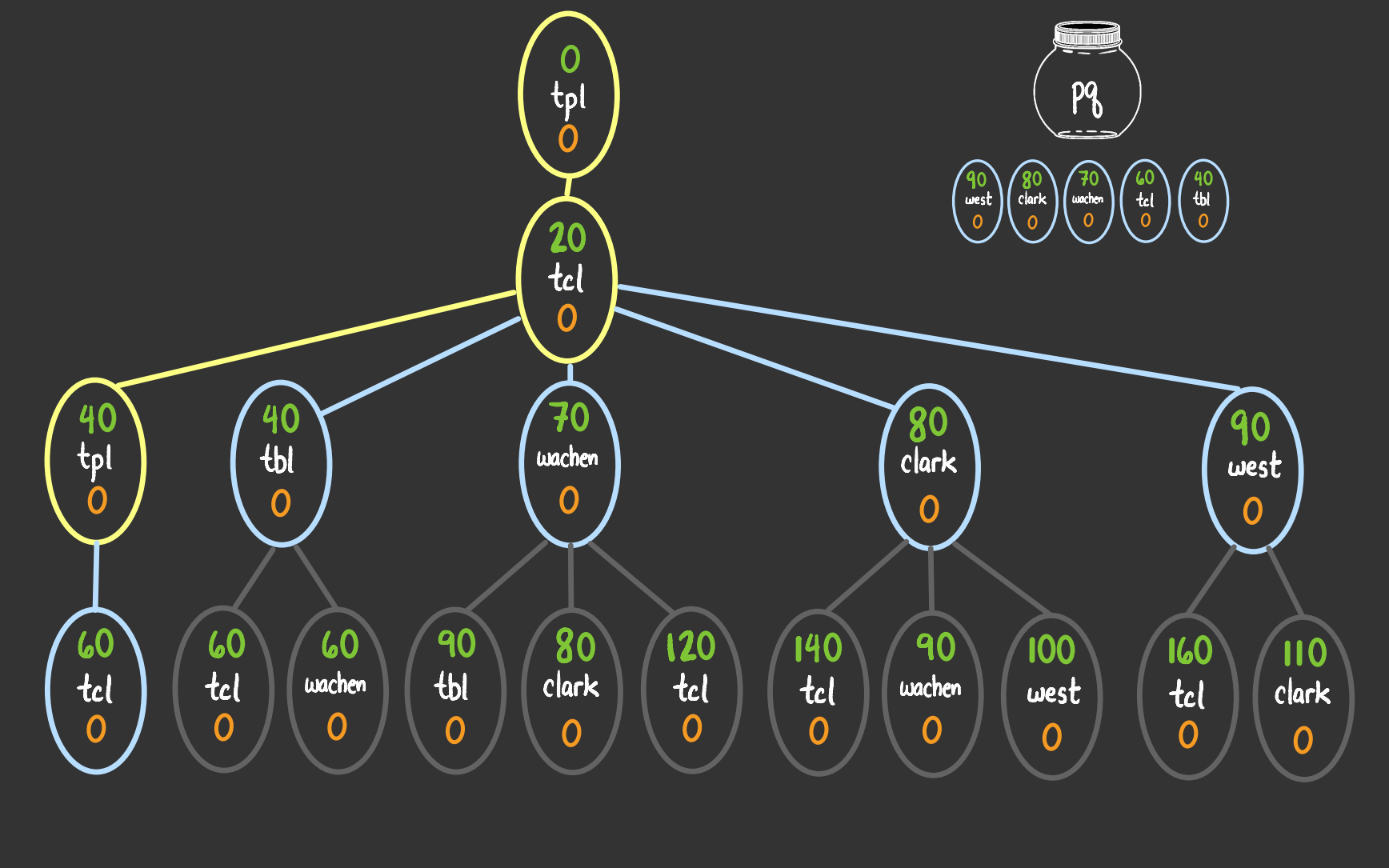

0
tpl
0
pq
20
tcl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

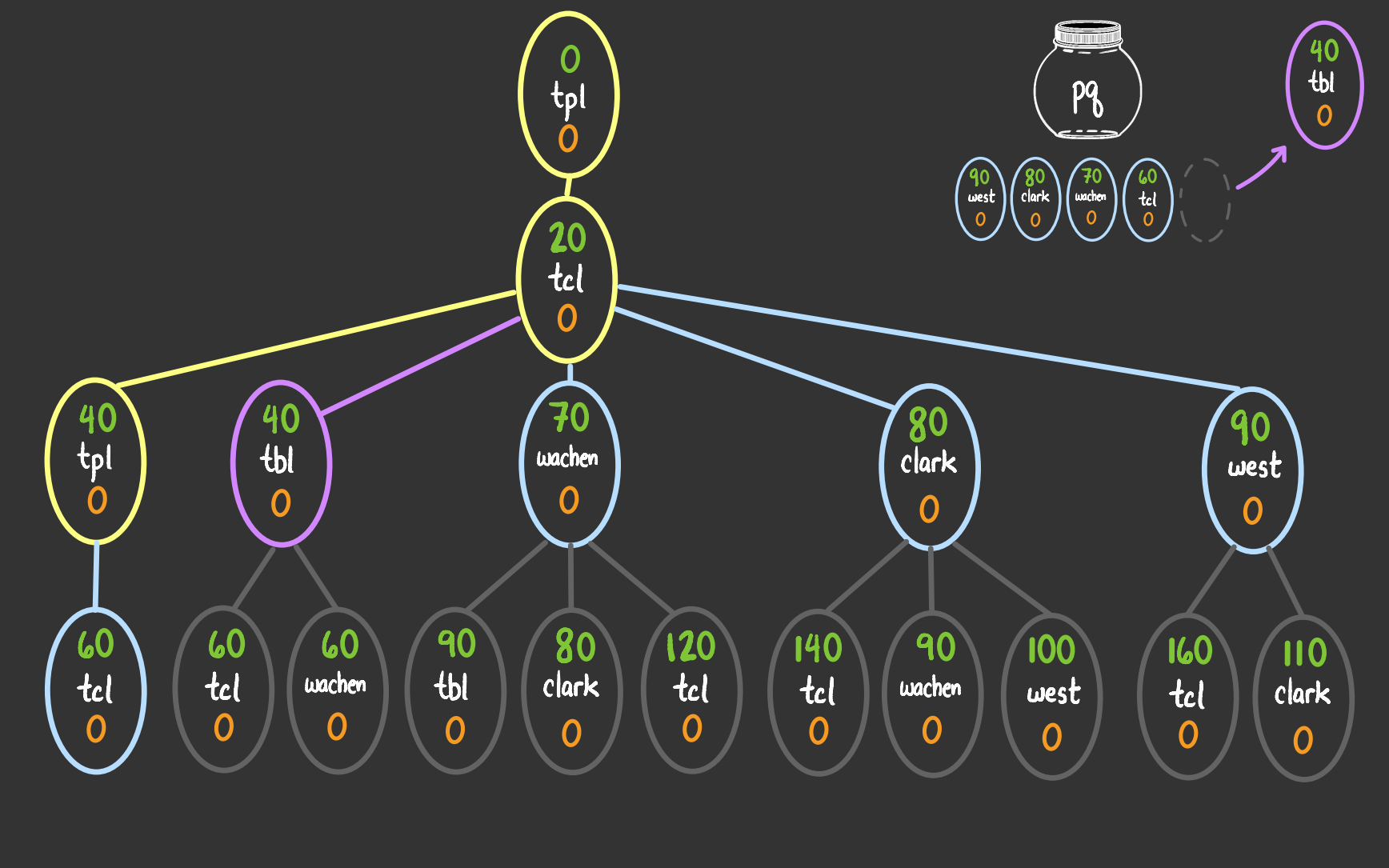

0
tpl
0
20
tcl
0
pq
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

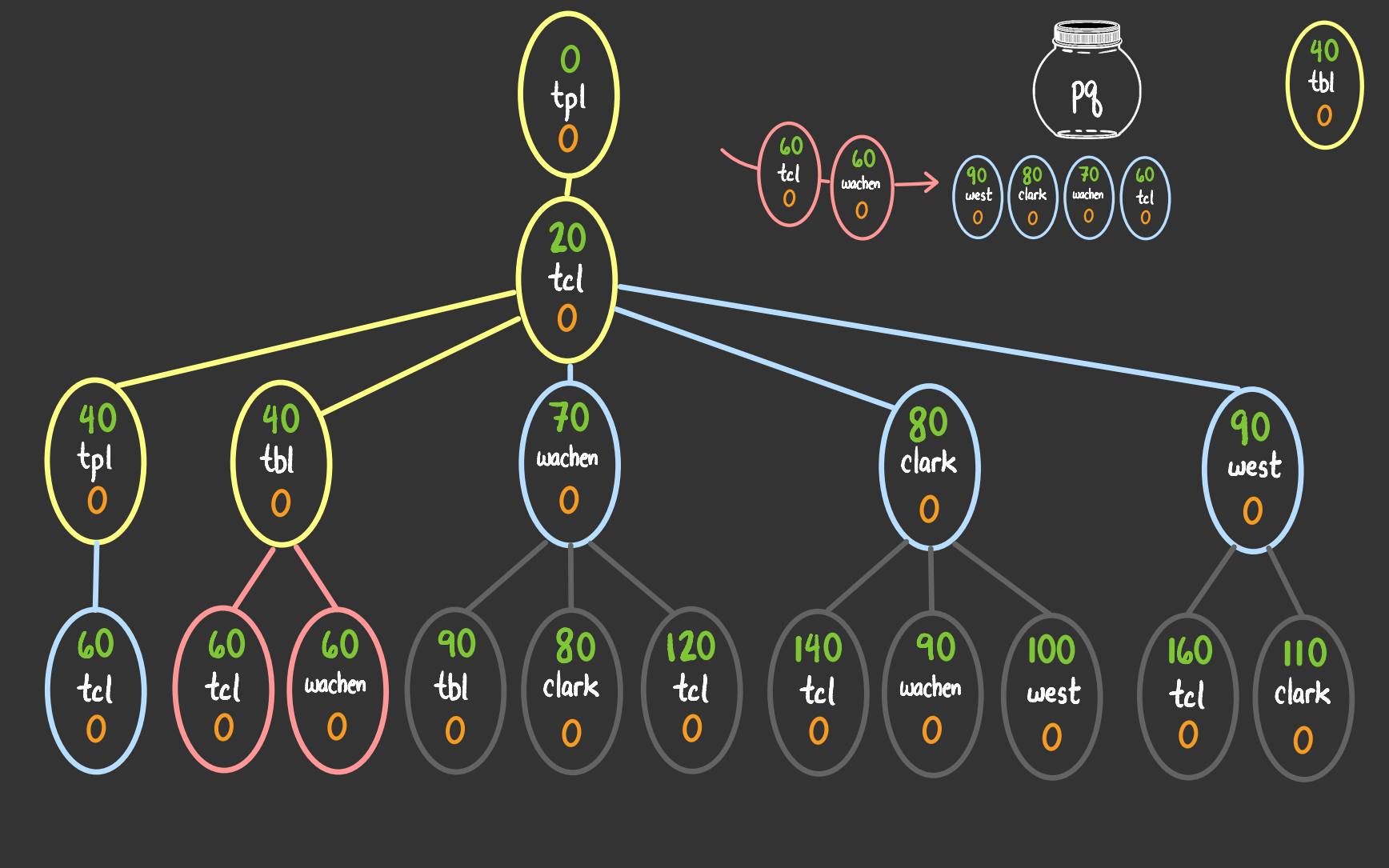

0
tpl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
pq
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

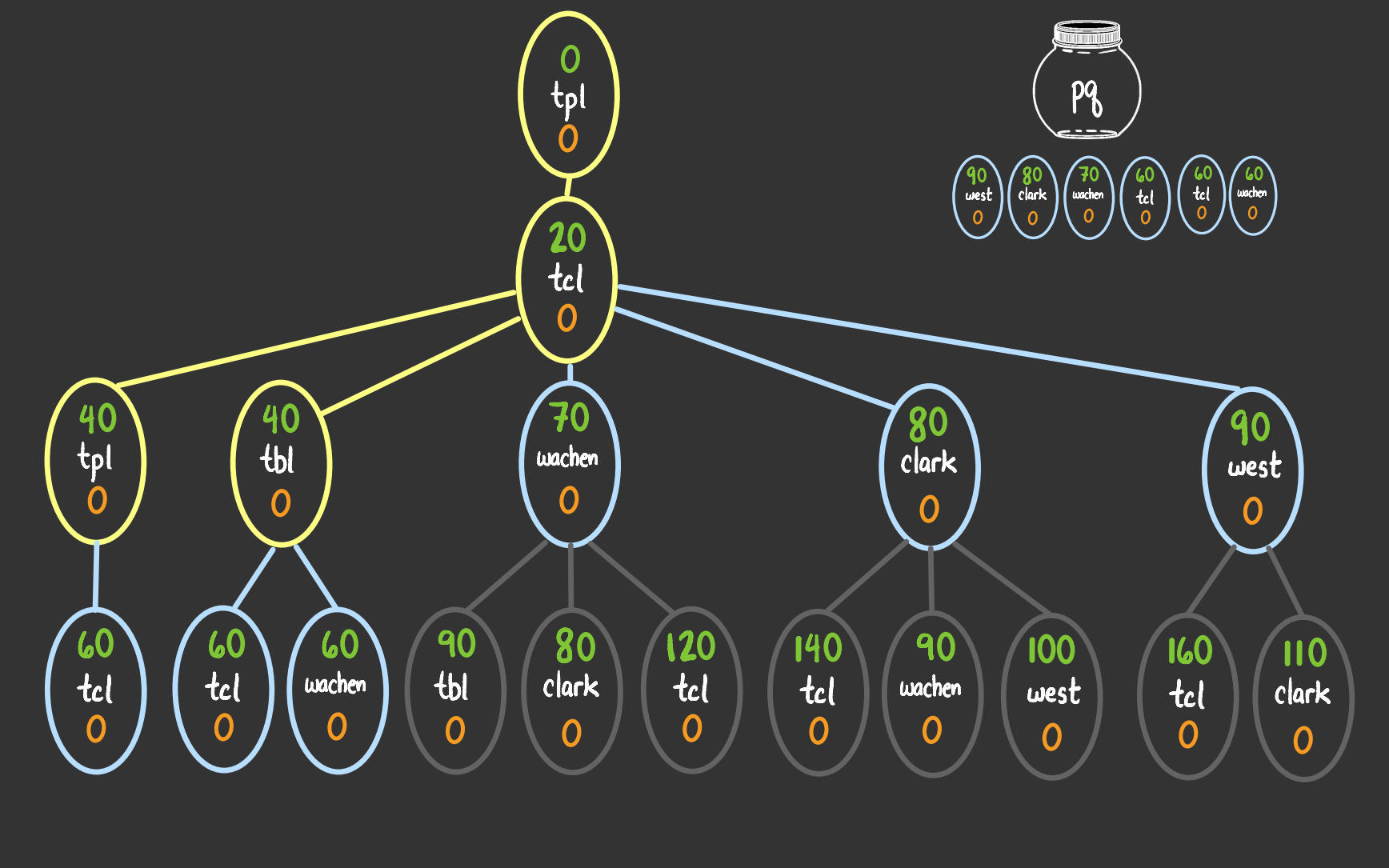

0
tpl
0
20
tcl
0
pq
90
west
0
80
clark
0
70
wachen
0
40
tbl
0
40
tpl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

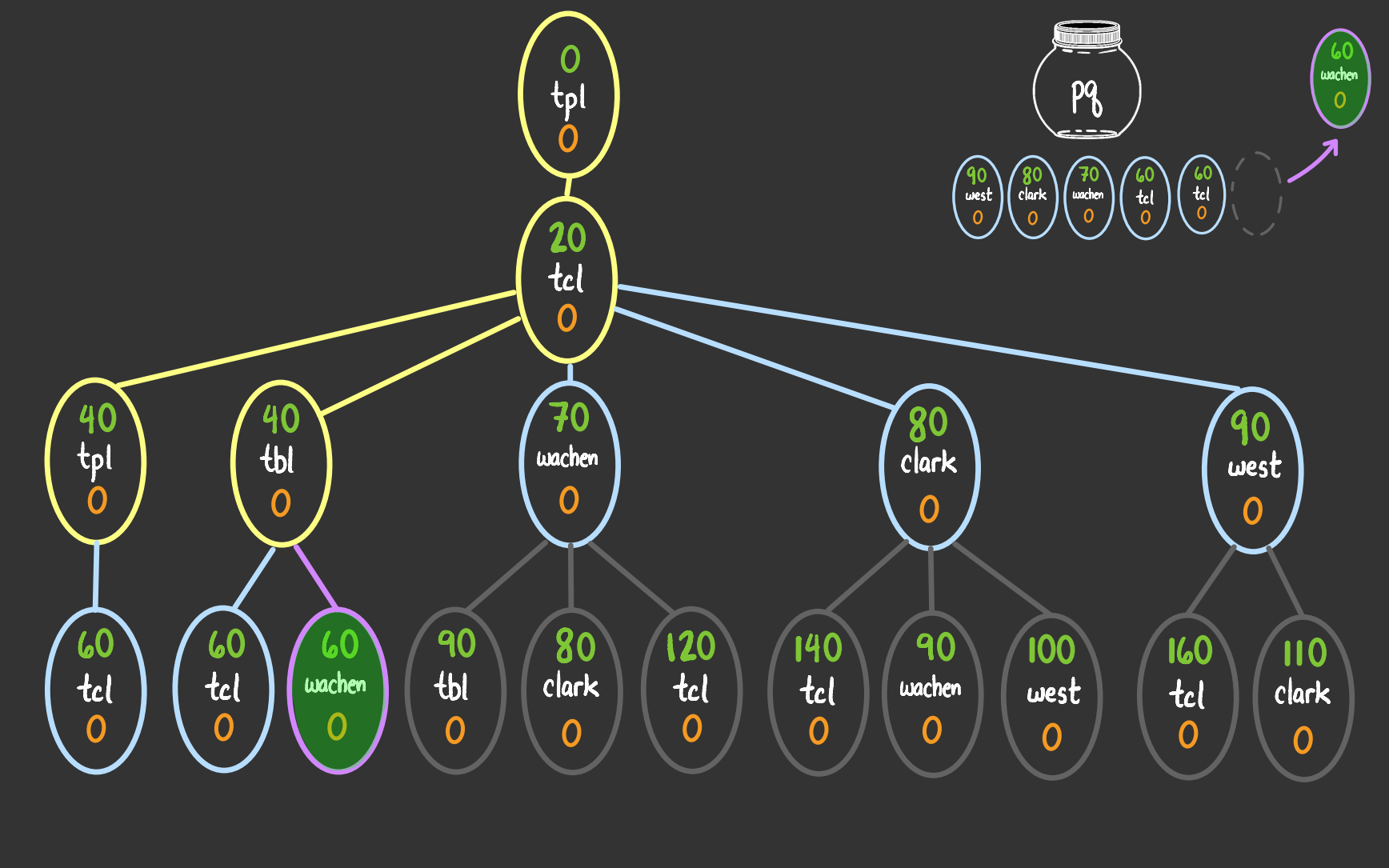

what happens next?
0 tpl 0
20 tcl 0
60 tcl 0
pq
90 west 0
80 clark 0
70 wachen 0
40 tbl 0
your answer here
40 tpl 0
40 tpl 0
40 tbl 0
70 wachen 0
80 clark 0
90 west 0
60 tcl 0
60 tcl 0
60 wachen 0
90 tbl 0
80 clark 0
120 tcl 0
140 tcl 0
90 wachen 0
100 west 0
160 tcl 0
110 clark 0

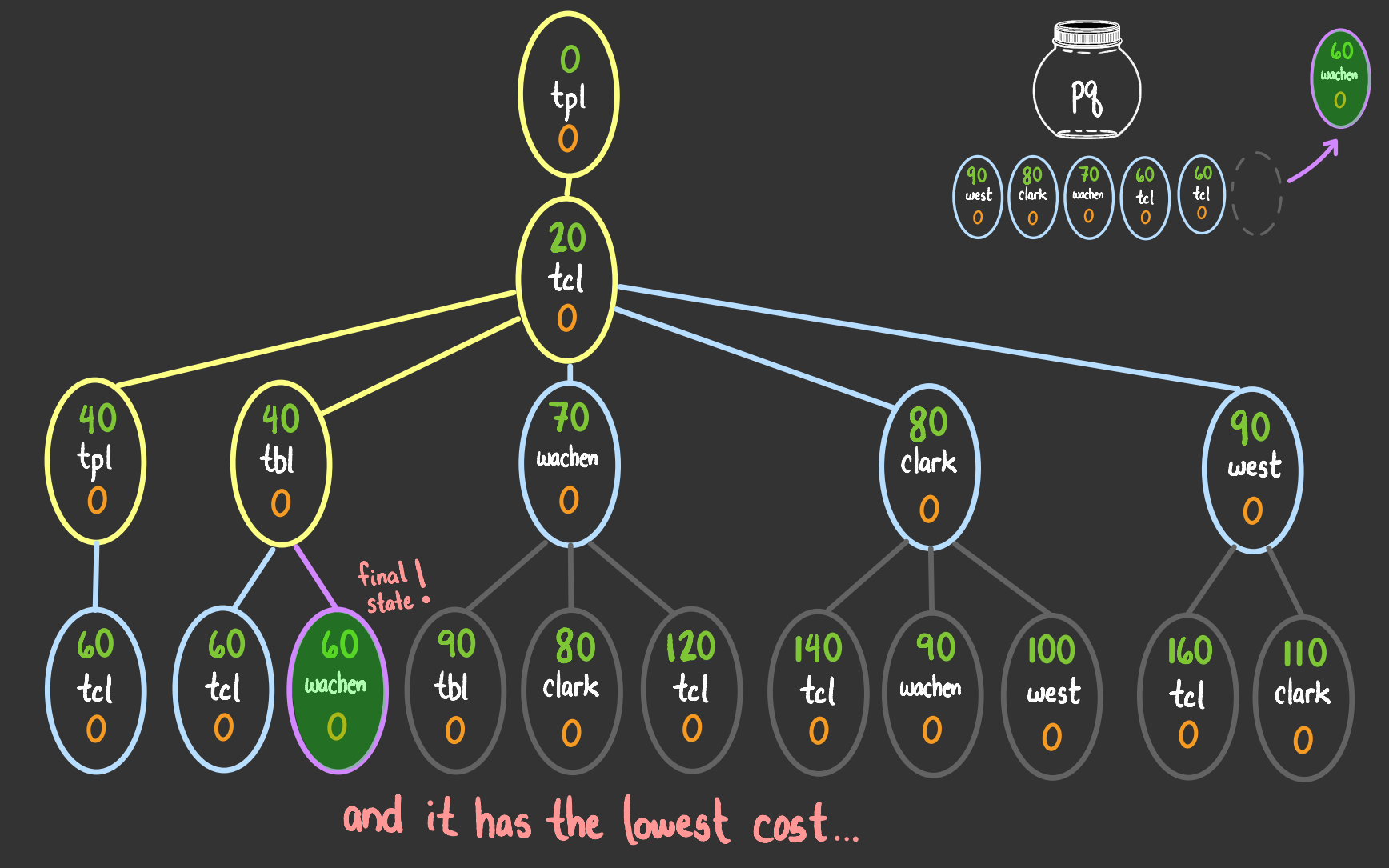

0
tpl
0
pq
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
40
tbl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0

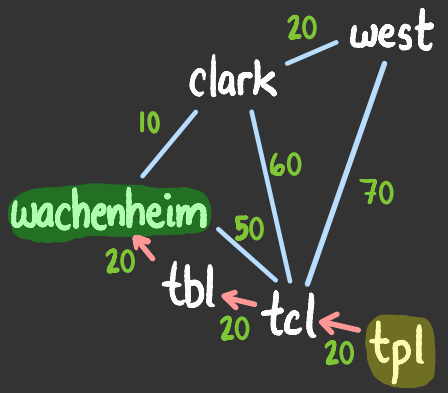

0
tpl
0
20
tcl
0
pq
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
40
tbl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0


0
tpl
0
pq
60
tcl
0
60
wachen
0
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
40
tbl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0


0
tpl
0
pq
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
60
tcl
0
60
wachen
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0


0
tpl
0
pq
60
wachen
0
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
60
tcl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0


0
tpl
0
pq
60
wachen
0
90
west
0
80
clark
0
70
wachen
0
60
tcl
0
60
tcl
0
20
tcl
0
40
tpl
0
40
tbl
0
70
wachen
0
80
clark
0
90
west
0
60
tcl
0
60
tcl
0
60
wachen
0
final ! state .
90
tbl
0
80
clark
0
120
tcl
0
140
tcl
0
90
wachen
0
100
west
0
160
tcl
0
110
clark
0
and it has the lowest cost...



uniform cost search is guaranteed to find the optimal solution

why? if search node n_1 is visited before search node n_2, then $g(n_1) \leq g(n_2)$

(can be proven by contradiction)