• Types:

• QUEUE [G]

-- G: Formal generic parameter

• Functions (Operations):

• in: QUEUE [G] × G → QUEUE [G] // 新元素从队尾入队

• out: QUEUE [G] /→ QUEUE [G] // 队首元素出队

• head: QUEUE [G] /→ G // 返回队首值

• tail: QUEUE [G] /→ G // 返回队尾值

• query: QUEUE [G] × G → BOOLEAN // 查询特定元素是否在队中

• empty: QUEUE [G] → BOOLEAN

• new: QUEUE [G]

• Preconditions:

• out (q: QUEUE [G]) require not empty (q)

• head (q: QUEUE [G]) require not empty (q)

• tail (q: QUEUE [G]) require not empty (q)

• Axioms: For all x: G, q: QUEUE [G]

• tail (in (q, x)) = x

• if empty (q) then head (in (q, x)) = x

else head (in (q, x)) = head (q)

• if empty (q) then out (in (q, x)) = q

else out (in (q, x)) = in (out(q), x)

• empty (new)

(or: empty (new) = True)

• not empty (in (q, x))

(or: empty (in (q, x)) = False)

• if empty (q) then query (q, x) = False

else if head (q) = x then query (q, x) = True

else query (q, x) = query (out (q), x)

• Types:

• MAP [K, V]

-- K: Key

-- V: Value

• Functions (Operations):

• put: MAP [K, V] × K × V → MAP [K, V] // 插入一个键值对

• remove: MAP [K, V] × K /→ MAP [K, V] // 移除一个键值对

• item: MAP [K, V] × K /→ V // 取出一个键对应的值

• has: MAP [K, V] × K → BOOLEAN // 查询是否存在一个键

• new: MAP [K, V] // 创建新表

• empty: MAP [K, V] → BOOLEAN

• Preconditions:

• put (k: K, v: V, m: MAP [K, V]) require not has (m, k)

• remove (k: K, m: MAP [K, V]) require has (m, k)

• item (k: K, m: MAP [K, V]) require has (m, k)

• Axioms: For all k, k’: K, v, v’: V, m: MAP [K, V]

• if k’ = k then item (put (m, k, v), k’) = v

else item (put (m, k, v), k’) = item (m)

• if k’ = k then remove (put (m, k, v), k’) = m

else remove (put (m, k, v), k’) = put (remove (m, k’), k, v)

• if empty (m) then has (m, k) = False

• if k’ = k then has (put (m, k, v), k’) = True

else has (put (m, k, v), k’) = has (m, k’)

• empty (new)

(or: empty (new) = True)

• not empty (put (m, k, v))

(or: empty (put (m, k, v)) = False)