- The container in which we store key - value pairs, and where a key can have multiple associated values, is called a **ADT MultiMap**.
  - Domain of ADT MultiMap:

$\mathcal{MM} = \{mm | mm$ is a Multimap with TKey, TValue pairs$\}$

- init (mm)
  - **descr:** creates a new empty multimap
  - **pre:** true
  - **post:** $mm \in \mathcal{MM}$, mm is an empty multimap
- destroy(mm)
  - **descr:** destroys a multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** the multimap was destroyed
- add(mm, k, v)
  - **descr:** add a new pair to the multimap
  - **pre:** $mm \in \mathcal{MM}$, $k - TKey$, $v - TValue$
  - **post:** $mm' \in \mathcal{MM}$, $mm' = mm \cup < k, v >$
- remove(mm, k, v)
  - **descr:** removes a key value pair from the multimap
  - **pre:** $mm \in \mathcal{MM}$, $k - TKey$, $v - TValue$
  - **post:** $remove \leftarrow$

$$\begin{cases} true, & \text{if } < k, v > \in mm, mm' \in \mathcal{MM}, mm' = mm - < k, v > \\ false, & \text{otherwise} \end{cases}$$

- search(mm, k, l)
  - **descr:** returns a list with all the values associated to a key
  - **pre:** $mm \in \mathcal{MM}$, $k - TKey$
  - **post:** $l \in \mathcal{L}$, l is the list of values associated to the key k. If k is not in the multimap, l is the empty list.

- iterator(mm, it)
  - **descr:** returns an iterator over the multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** $it \in \mathcal{I}$, *it* is an iterator over *mm*, the current element from *it* is the first pair from *mm*, or, *it* is invalid if *mm* is empty
  - **Obs:** the iterator for a MultiMap is similar to the iterator for other containers, but the *getCurrent* operation returns a <key, value> pair.
- size(mm)
  - **descr:** returns the number of pairs from the multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** $size \leftarrow$ the number of pairs from mm
- Other possible operations:
- keys(mm, s)
  - **descr:** returns the set of all keys from the multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** $s \in \mathcal{S}$, *s* is the set of all keys from *mm*
- values(mm, b)
  - **descr:** returns the bag of all values from the multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** $b \in \mathcal{B}$m *b* is a bag with all the values from *mm*

- pairs(mm, b)
  - **descr:** returns the bag of all pairs from the multimap
  - **pre:** $mm \in \mathcal{MM}$
  - **post:** $b \in \mathcal{B}$, b is a bag with all the pairs from mm
- We can have a MultiMap where we can define an order (a relation) on the set of possible keys. However, if a key has multiple values, they can be in any order (we order the keys only, not the values) $\Rightarrow$ **ADT SortedMultiMap**

- The only change in the interface is for the *init* operation that will receive the *relation* as parameter.

- For a sorted MultiMap, the iterator has to iterate through the pairs in the order given by the *relation*, and the operations *keys* and *pairs* return SortedSet and SortedBag.