Problem

- **Implement the SortedMultiMap ADT**

- use a singly linked representation with dynamic allocation

- use ADT List (as a second container - if needed)

Example:

a multimap for the translation of different English words in Romanian

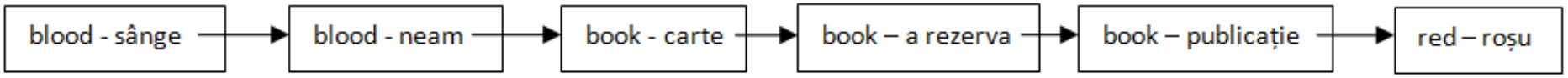
- book – carte, a rezerva, publicație
- red – roșu
- blood – sânge, neam

SMM. Representation

- **Representation 1**:

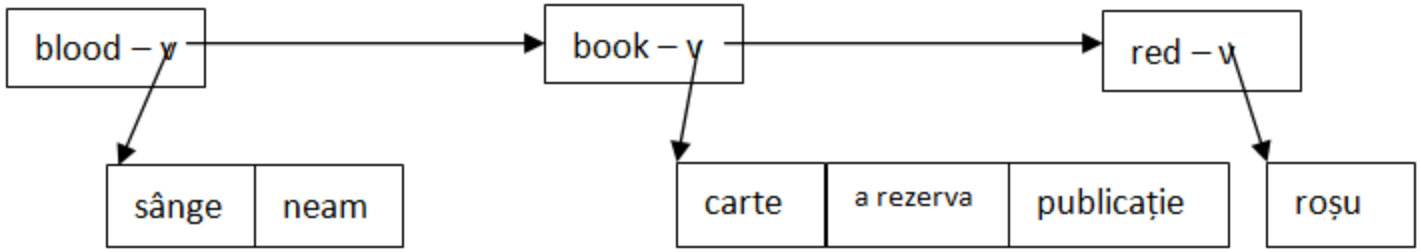
Singly linked list of <key, value> pairs. There might be multiple nodes with the same key, they will be placed one after the other (since the nodes are sorted based on the keys).

- sorted based on the keys).



- **Representation 2**

Singly linked list of <key, list of values> pairs. The keys are unique and sorted.



Representation 2 - SMM

TElem:
k: TKey
vl: List

Node:
info: TElem
next: ↑Node

SMM:
head: ↑Node
R: Relation

$$R(k_1, k_2) = \begin{cases} true, if\ "k_1 \leq k_2"\ (k_1 comes\ before\ k_2) \\ false, otherwise \end{cases}$$

Iterator

Iterator operations:

init, valid, next, getCurrent (returns a <key, value> pair).

Example

Printing the elements of a SMM using the iterator:

```
Subalgorithm print(smm) is:
        iterator(smm, it)
        while valid(it) execute:
                getCurrent(it, <k,v>)
                @print k and v
                next(it)
        end-while
end-subalgorithm
```

The print subalgorithm looks in the same way independently of the representation of the iterator and the representation of the map!

Representation 2 – SMM Iterator

TElem:
k: TKey
vl: List

Node:
info: TElem
next: ↑Node

SMM:
head: ↑Node
R: Relation

$$R(k_1, k_2) = \begin{cases} true, if\ "k_1 \leq k_2"\ (k_1 comes\ before\ k_2) \\ false, otherwise \end{cases}$$

IteratorSMM:
smm: SMM
current: ↑Node
itL: IteratorList

Subalgorithm next(it) is: