

Dicționar (MAP)

Observații

1. Elementele din dicționar sunt perechi de forma (**cheie**, **valoare**). Dicționarele păstrează elemente în așa fel încât ele să poată fi ușor localizate folosind **chei**.
2. Spre exemplu, un dicționar poate păstra conturi bancare: fiecare cont este un obiect identificat printr-un număr de cont (considerat **cheia** elementului) și informații adiționale (numele și adresa deținătorului contului, informații despre depozite, etc). Informațiile adiționale vor fi considerate ca fiind **valoarea** elementului.
3. Implementarea unui dicționar (SD aleasă pentru implementare) trebuie să ofere un mecanism eficient de regăsire a valorilor pe baza cheilor.
4. Într-un dicționar cheile sunt **unice**.
5. În general, o **cheie** are o unică **valoare** asociată. Dacă o cheie poate avea mai multe valori asociate => Multi-dicționar (**MultiMap**)

Dăm în continuare specificația Tipului Abstract de Date **Dicționar**.

domeniu

$\mathcal{D} = \{d \mid d \text{ este un dicționar cu elemente } e = (c, v), c \text{ de tip } T\text{Cheie}, v \text{ de tip } T\text{Valoare}\}$

operații (interfața TAD-ului Dicționar)

crează(d)

pre: true

post: $d \in \mathcal{D}$, d este dicționarul vid (fără elemente)

adaugă(d, c, v)

pre: $d \in \mathcal{D}, c \in T\text{Cheie}, v \in T\text{Valoare}$,

post: $d' \in \mathcal{D}, d' = d + (c, v)$ (se adaugă în dicționar perechea (c, v))

caută(d, c, v)

pre: $d \in \mathcal{D}, c \in T\text{Cheie}$

post: *caută* = adevărat

fals

dacă $(c, v) \in d$, caz în care $v \in T\text{Valoare}$ e
valoarea asociată cheii c

în caz contrar, caz în care $v = 0_{T\text{Valoare}}$

șterge(d, c, v)

pre: $d \in \mathcal{D}$, $c \in \text{TCheie}$

post: $v \in \text{TValoare}$

perechea (c, v) este ștearsă din dicționar, dacă $c \in d$

$v = 0_{\text{TValoare}}$ în caz contrar

dim(d)

pre: $d \in \mathcal{D}$

post: *dim* = dimensiunea dicționarului d (numărul de elemente) $\in \mathcal{N}^*$

vid(d)

pre: $d \in \mathcal{D}$

post: *vid* = adevărat în cazul în care d e dicționarul vid
fals în caz contrar

chei(d, m)

pre: $d \in \mathcal{D}$

post: $m \in \mathcal{M}$, m este mulțimea cheilor din dicționarul d

valori(d, c)

pre: $d \in \mathcal{D}$

post: $c \in \text{Col}$, c este colecția valorilor din dicționarul d

perechi(d, m)

pre: $d \in \mathcal{D}$

post: $m \in \mathcal{M}$, m este mulțimea perechilor (cheie, valoare) din dicționarul d

iterator(d, i)

{se creează un iterator pe dicționarul d}

pre: $d \in \mathcal{D}$

post: $i \in I$, i este iterator pe dicționarul d

distruge(d)

pre: $d \in \mathcal{D}$

post: dicționarul d a fost 'distrus' (spațiul de memorie alocat a fost eliberat)

Modalități de implementare ale dicționarelor:

1. tablouri (dinamice);
2. liste înlanțuite;
3. tabele de dispersie;
4. arbori binari.

Observații

1. Multi-dicționar (**MultiMap**)

- **TValoare=TLista** (o cheie are o listă de valori asociate **TElement**)
- Operație din interfața TAD Dicționar a cărei specificație se modifică
 - *șterge*(d, c, v)

pre: $d \in \mathcal{D}$, $c \in Tcheie$, $v \in TElement$

post: $l \in TValoare$

perechea (c, v) este ștearsă din dicționar, dacă $c \in d$
l e noua listă a valorilor asociate cheii c (posibil vidă)

2. Dicționar ordonat/sortat (**SortedMap**)

- **TCheie=TComparabil**
- Este definită o relație de ordine între chei $\mathfrak{R} \subseteq TCheie \times TCheie$
- Nu se modifică interfață
- **Cerință** – operațiile **iterator** și **perechi** returnează elementele în ordine în raport cu relația \mathfrak{R}

3. Multi-dicționar ordonat/sortat (**Sorted MultiMap**)