Dicţionar (MAP)

Observaţii

1. Elementele din dicţionar sunt perechi de forma **(cheie, valoare)**. Dicţionarele păstrează elemente în aşa fel încât ele să poată fi uşor localizate folosind **chei**.
2. Spre exemplu, un dicţionar poate păstra conturi bancare: fiecare cont este un obiect identificat printr-un număr de cont (considerat **cheia** elementului) şi informaţii adiţionale (numele şi adresa deţinătorului contului, informaţii despre depozite, etc). Informaţiile adiţionale vor fi considerate ca fiind **valoarea** elementului.
3. Implementarea unui dicţionar (SD aleasă pentru implementare) trebuie să ofere un mecanism eficient de regăsire a valorilor pe baza cheilor.
4. Într-un dicţionar 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** | **d** este un dicţionar cu elemente **e = (c, v), c** de tip **TCheie, v** de tip **TValoare**}

operaţii (**interfaţa** TAD-ului Dicţionar)

creează(d)
pre: true
post:d$\in\mathcal{D}$, d este dicţionarul vid (fără elemente)

adaugă(d, c, v)
pre: d$\in\mathcal{D}$, c$\in$**TCheie,** v$\in$**TValoare,**
post: d'$\in\mathcal{D}$, d'=d+(c, v) (se adaugă în dicţionar perechea (c, v))
{dacă există deja cheia în dicționar, înlocuiește valoarea asociată cheii și se poate returna vechea
{valoare. Daca nu exista cheia, adauga perechea si se poate returna $0_{TValoare}$}

caută(d, c, v)
pre: d$\in\mathcal{D}$, c$\in$**TCheie**
post: ***caută*****=** adevărat dacă (c,v)$\in$d, caz în care v$\in$**TValoare** e valoarea asociată cheii c
fals în caz contrar, caz în care v=$0_{TValoare}$

şterge(d, c, v)

pre: d∈$\mathcal{D}$, c∈**TCheie**

post: v∈**TValoare**

perechea (c, v) este ştearsă din dicţionar, dacă c∈ d

v= $0_{TValoare}$ în caz contrar

dim(d)

pre: d∈$\mathcal{D}$

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

vid(d)

pre: d∈$\mathcal{D}$

post: ***vid***= adevărat în cazul în care d e dicţionarul vid

fals în caz contrar

chei(d, m)

pre: d∈$\mathcal{D}$

post: m∈$\mathcal{M}$, m este mulţimea cheilor din dicţionarul d

valori(d, c)

pre: d∈$\mathcal{D}$

post: c∈$\mathcal{C}ol$, c este colecţia valorilor din dicţionarul d

perechi(d, m)

pre: d∈$\mathcal{D}$

post: m∈$\mathcal{M}$, m este mulţimea perechilor (cheie, valoare) din dicţionarul d

iterator(d, i)

{se creează un iterator pe dicţionarul d}

pre: d∈ $\mathcal{D}$

post:i∈ $\mathcal{I}$, i este iterator pe dicţionarul d

distruge(d)

pre: d∈ $\mathcal{D}$

post: dicţionarul d a fost 'distrus' (spaţiul de memorie alocat a fost eliberat)

Modalităţi de implementare ale dicționarelor:

- tablouri (dinamice);
- liste înlănţuite;
- tabele de dispersie;
- arbori binari.

Observaţii

1. Multi-dicționar (**MultiMap**)
   - O cheie are o listă de valori asociate
   - Operație din interfața TAD Dicționar a cărei specificație se modifică
     - ***şterge***(d, c, v)

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

       post: d'∈$\mathcal{D}$

       perechea (c, v) este ştearsă din dicţionar, dacă c∈ d

2. Dicționar ordonat/sortat (**SortedMap**)
   - **TCheie**=**TComparabil**
   - Este definită o relație de ordine între chei $\Re \subseteq TCheie \times TCheie$
   - Nu se modifică interfața
   - **Cerință** – operațiile **iterator** și **perechi** returnează elementele în ordine în raport cu relația $\Re$

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