

ADT Map/ Associative Array

IF2110 – Algoritma dan Struktur Data
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Map – Definisi

Map adalah kumpulan pasangan/*binding* $\langle key, value \rangle$, dengan nilai *key* bersifat unik di dalam kumpulan tsb.

Dikenal juga sebagai *associative array*, *symbol table*, atau *dictionary*.

Operasi terhadap Map:

- Penambahan sebuah pasangan baru
- Penghapusan suatu pasangan
- Modifikasi nilai *value* dari suatu pasangan
- Pencarian nilai *value* dari suatu *key* tertentu.

Map – Definisi Fungsional

Jika diberikan M adalah **Map** dengan elemen pasangan $\langle K, V \rangle$

CreateMap: $\rightarrow M$ { Membuat sebuah Map kosong }

isEmpty: $M \rightarrow \text{boolean}$ { Tes terhadap M : true jika M kosong,
false jika M tidak kosong }

set: $K \times V \times M \rightarrow M$ { Menambahkankan pasangan (K, V) ke M jika
belum ada elemen dengan nilai key= K pada M ,
atau mengubah nilai value dari pasangan
dengan nilai key= K menjadi V }

unset: $K \times M \rightarrow M$ { Menghapus pasangan dengan nilai key= K
dari M }

find: $K \times M \rightarrow V$ { Mengembalikan value dari pasangan dengan
nilai key= K }

Axiomatic Semantics (fungsional)

- 1) $\text{new}()$ returns a map
- 2) $\text{find}(k, \text{set}(k, v, M)) = v$
- 3) $\text{find}(k, \text{set}(j, v, M)) = \text{find}(k, M)$ if $k \neq j$
- 4) $\text{unset}(k, \text{new}()) = \text{new}()$
- 5) $\text{unset}(k, \text{set}(k, v, M)) = \text{unset}(k, M)$
- 6) $\text{unset}(k, \text{set}(j, v, M)) = \text{set}(j, v, \text{unset}(k, M))$ if $k \neq j$

dengan k dan j adalah key, v adalah value, dan M adalah map.

Map – Implementasi

Jika jumlah elemen sedikit, dapat digunakan *association list* dalam bentuk array dengan elemen pasangan $\langle key, value \rangle$.

Jika nilai *key* terbatas pada selang nilai integer tertentu, dapat digunakan *direct addressing* terhadap tabel: pasangan $\langle key, value \rangle$ disimpan dengan memberi nilai *value* pada indeks ke-*key*.

Jika nilai *key* memiliki rentang yang besar namun data hanya sedikit, *direct addressing* menjadi boros memori.

- Perlu dilakukan pemetaan *key* ke *address*/indeks.
- Implementasi yang paling umum digunakan adalah menggunakan *hash table*.

ADT Map (bandingkan dengan Set!)

KAMUS UMUM

constant CAPACITY: integer = ... { *Banyaknya elemen maksimum* }

constant VAL_UNDEF: ElType = ...

type KeyType: ...

type ElType: ...

type MapEntry: < key: KeyType, value: ElType > { *elemen map* }

type Map:

< buffer: array [0..CAPACITY-1] of MapEntry, { *array penyimpan elemen map* }
length: integer > { *jumlah elemen map* }