Interfaccia Map

Problema

- Dovete tenere traccia delle disponibilita’ a magazzino di un negozio di bulloni
- I bulloni sono identificati da un codice numerico
- Ci sono 10 tipi di bulloni
- I codici validi sono 1,..10

Soluzione

- Banale!
- Si usa un’array in cui l’indice e’ dato dal codice pezzo –1

- Si occupa esattamente la memoria richiesta
- La disponibilita’ a magazzino e’ accessibile con complessita’ O(1)

Cosa succede se…

- I tipi di bulloni diversi sono 100
- I codici sono numerici
- I codici validi non sono compattati tra 1 e 100
- ma invece 100 diversi compresi tra 1 e 1.000.000.000

Soluzioni?

- Uso un array con 1.000.000.000 posizioni
  - Ricerca in O(1), ma…
  - ne riempo solo 100
  - spreco del 99.99999%
- Uso una lista
  - Uso solo memoria effettivamente richiesta, ma…
  - ricerca con complessita' O(N)
    - se le chiavi sono effettivamente tutte comparabili, O(logN) in termini di confronti
    - BinarySearch su lista
- Scadenti entrambe

java.util.Map

- Interfaccia
- Definisce un servizio che permette di associare (chiave, oggetto) in modo che:
  - Data la chiave e' possibile cercare l'oggetto con complessita' quasi O(1)
  - E' possibile controllare la presenza di una chiave nella mappa
  - E' possibile controllare la presenza di un oggetto nella mappa
  - E' possibile ottenere il *Set* delle chiavi
  - E' possibile ottenere la *Collection* degli oggetti

Javadocs

- void **clear**()
  - Removes all mappings from this map (optional operation).
- boolean **containsKey**(Object key)
  - Returns true if this map contains a mapping for the specified key.
- boolean **containsValue**(Object value)
  - Returns true if this map maps one or more keys to the specified value.
- Set **entrySet**()
  - Returns a set view of the mappings contained in this map.

Javadocs (Cont.)

- boolean **equals**(Object o)
  - Compares the specified object with this map for equality.
- Object **get**(Object key)
  - Returns the value to which this map maps the specified key.
- int **hashCode**()
  - Returns the hash code value for this map.
- boolean **is Empty**()
  - Returns true if this map contains no key-value mappings.

Javadocs

- Set **keySet**()
  - Returns a set view of the keys contained in this map.
- Object **put**(Object key, Object value)
  - Associates the specified value with the specified key in this map (optional operation).
- void **putAll**(Map t)
  - Copies all of the mappings from the specified map to this map (optional operation).
- Object **remove**(Object key)
  - Removes the mapping for this key from this map if it is present (optional operation).

Javadocs

- int **size**()
  - Returns the number of key-value mappings in this map.
- Collection **values**()
  - Returns a collection view of the values contained in this map.

Classe Concreta?

- java.util.Hashtable
  - Implementa l’interfaccia Map
  - aggiunge alcuni metodi con semantica simile a metodi presenti nell’interfaccia Map ma risalenti a versioni precedenti
  - retrofittata per Collections framework
- java.util.HashMap
  - Implementa l’interfaccia Map

Limiti?

- Non e’ possibile avere chiavi uguali per oggetti diversi
- Non e’ possibile avere una chiave nulla
- La complessita’ e’ approssimativamente O(1)
- L’occupazione di memoria e’ superiore a quella di un array che contenga gli elementi
  - si devono memorizzare anche le chiavi

Implementazione?

- Una funzione di generazione di una posizione a partire dall'oggetto chiave
  - funzione di hash
- Un'array di liste di coppie (key,value)