UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

DISIM
Dipartimento di Ingegneria
e Scienze dell'Informazione
e Matematica

Informatica
@UnivAQ

Laboratorio di Algoritmi e Strutture Dati
a.a. 2022/2023

La struttura dati Heap binario

Giovanna Melideo
Università degli Studi dell'Aquila
DISIM

Heap binario (Richiami)

- Struttura dati associata ad un **insieme totalmente ordinato** S
- Albero binario radicato con le seguenti proprietà:
  1. Un heap binario è **quasi completo**, ovvero completo fino al penultimo livello, con tutte le foglie sull'ultimo livello 'compattate' a sinistra
  2. Gli elementi di S sono memorizzati nei nodi dell'albero (ogni nodo v memorizza uno e un solo elemento di S, denotato con chiave(v)∈S)
  3. Ordinamento a heap (**proprietà di ordinamento parziale dell'heap**)
     - **min-heap:** per ogni nodo v dell'albero diverso dalla radice, chiave(padre(v)) **≤** chiave(v)
     - **max-heap:** per ogni nodo v dell'albero diverso dalla radice, chiave(padre(v)) **≥** chiave(v).

Heap binario (continua)

- La “quasi” completezza garantisce che
  - l’altezza di un heap con n nodi è **Θ(log n)**
  - le operazioni fondamentali sugli heap vengono eseguite in un tempo che è al più proporzionale all’altezza dell’heap e, quindi, richiedono un tempo O(log n)
  - Poiché le foglie sull’ultimo livello sono compattate a sinistra è possibile rappresentare l’albero implicitamente tramite un **array posizionale H**
    - **La radice occupa H[1]**
    - Se i è l’indice di un dato nodo, gli indici del padre parent[i]= i/2, left[i]=2i e right[i]= 2i+1

L’interfaccia BinaryHeap

```
public interface BinaryHeap<T> {

        public boolean isEmpty();

        public int size();

        public void insert( T x );

        public T findMax( );

        public T deleteMax( );

        public void heapify( );

        public void clear( );

}
```

```
/**
  * isEmpty(): Should be implemented to return true if the binary
  * heap is empty and false otherwise.
  */

/**
  * size(): Should be implemented to return the number of elements
  * in the binary heap.
  */

/**
 * insert(): Insert the item x into the array, maintaining heap order.
 * Duplicates are allowed.
 */

/**
 * findMax(); Find the greatest item in the heap.
 * @return the greatest item, or throw an exception if empty.
 */

/**
 * deleteMax(): Remove the greatest item from the priority queue.
 * @return the greatest item, or throw an exception if empty.
 */

/**
 * heapify(): Establish heap order property from an arbitrary
 * arrangement of items. Runs in linear time.
 */

/**
 * clear(): Make the heap logically empty.
 */
```

Implementazione basata su array

- L'heap binario è rappresentato mediante un array H di oggetti di tipo Comparable<>
- Il numero di elementi inseriti (dimensione logica) è memorizzato in un contatore **`size`**
- le chiavi sono memorizzate nelle locazioni H[1], …,H[**`size`**] (la locazione H[0] sarà usata come sentinella)
- H è ridimensionato con la tecnica del raddoppiamento/dimezzamento
  - **`rif. ArrayBinaryHeap.java`**

Implementazione basata su ArrayList

- L'heap binario è rappresentato mediante un oggetto di tipo ArrayList di oggetti di tipo Comparable.
- La dimensione logica coincide con quella fisica.
- È possibile fornire un criterio aggiuntivo di confronto tra elementi basato sull'uso di un comparatore.
  - **`rif. ArrayListBinaryHeap.java`**
- **`Nota:`** se i è l'indice di un dato nodo, `i*2 <= size` significa che il nodo è interno (ha almeno il figlio sinistro)

UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

DISIM
Dipartimento di Ingegneria
e Scienze dell'Informazione
e Matematica

Informatica
@UnivAQ

Domande?

Giovanna Melideo
Università degli Studi dell'Aquila
DISIM