
Implementazione di alberi

Definire una struttura dati

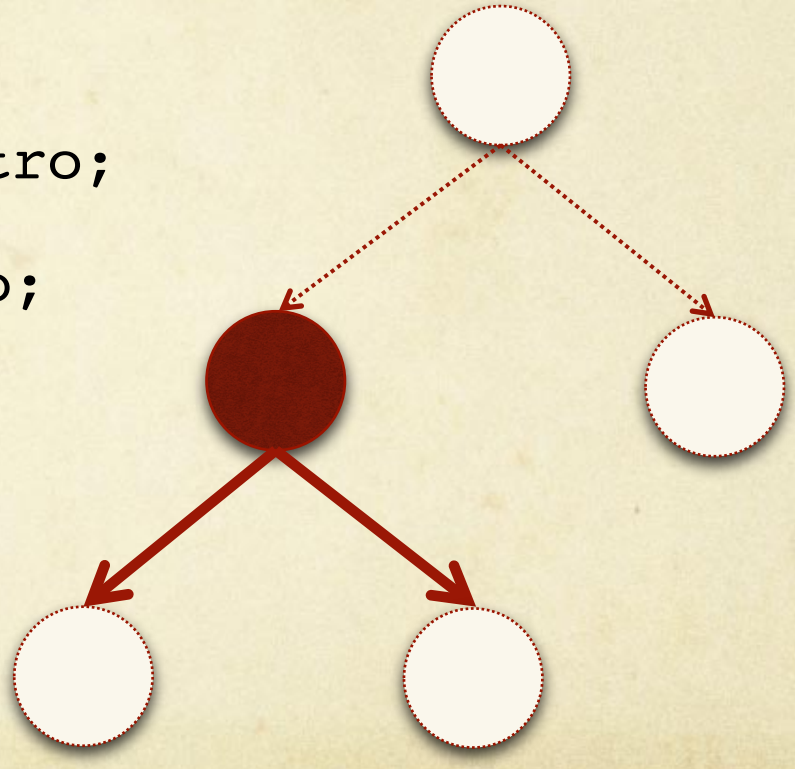
- Per definire una struttura dati in Java tramite l'utilizzo di reference, prima occorre definire "come sono fatti" gli elementi e poi definire la struttura dati.

Alberi

- Nella struttura dati “albero” gli elementi sono chiamati nodi
- Come è fatto un nodo di un albero binario?

Prima l'elemento...

```
class Entry<T>{

      T data;

      Entry<T> figlioSinistro;

      Entry<T> figlioDestro;

      …

}
```

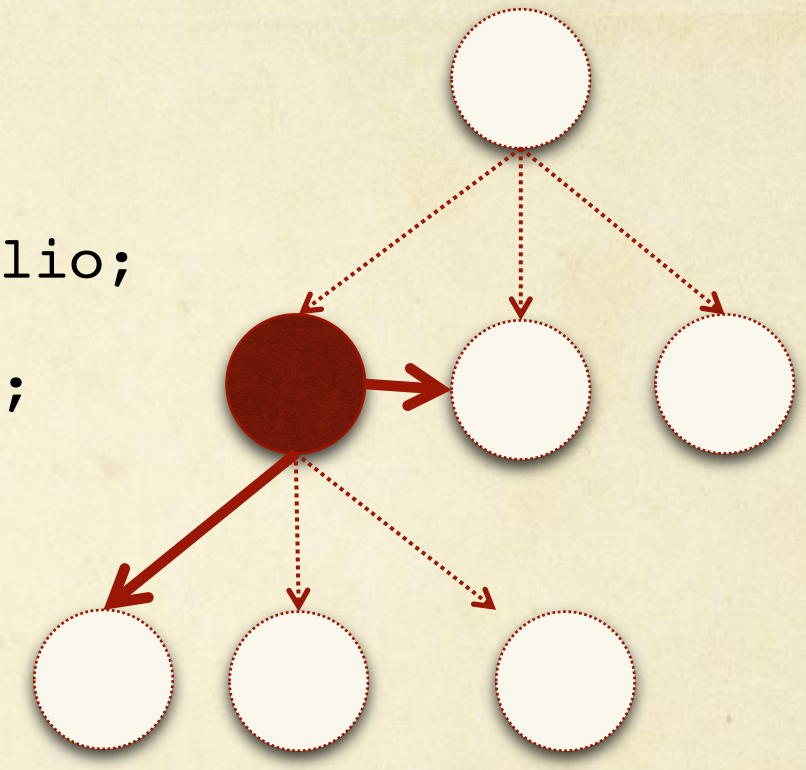


… poi la struttura

```
class Albero<T>{

     Entry<T> radice;

     …

}
```

Alberi n-ari

```
class Entry<T>{

      T data;

      Entry<T> primoFiglio;

      Entry<T> fratello;

      …

}

class Albero<T>{

      Entry<T> radice;

      …

}
```



Costruire un albero binario

- Metodo insert?

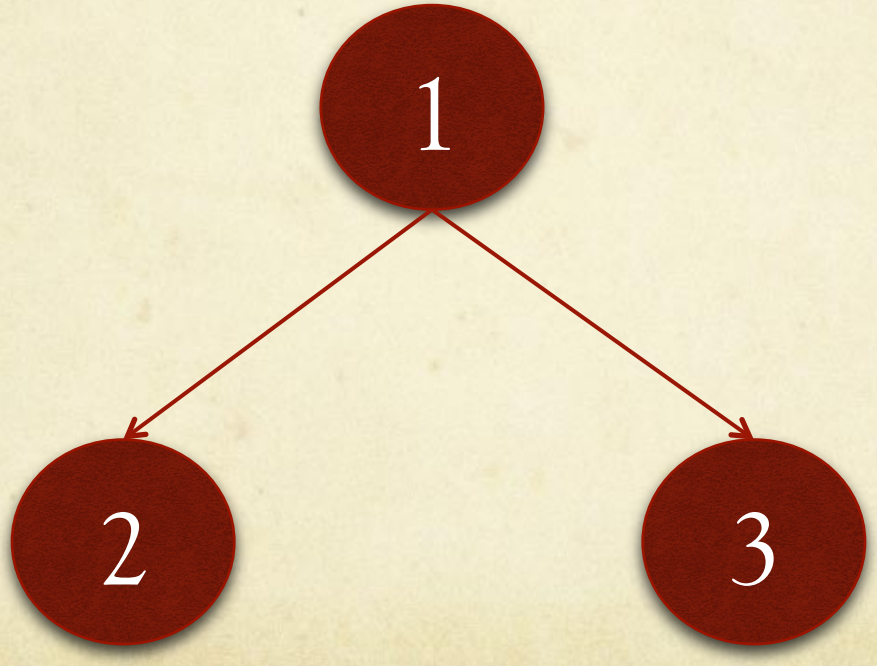
```
class Albero<T>{
        Entry<T> radice;
        …
        public void insert(Node n){
        …
        }
}
```

Esercizio

- Implementare il metodo insert dell'albero binario in modo tale che il nodo venga inserito (come foglia) in una posizione casuale. (Suggerimento: visitare l'albero a partire dalla radice; per ogni nodo incontrato lanciare una monetina per decidere se discendere per il sottoalbero destro o sinistro; se il sottoalbero scelto non esiste vi inserisco il nuovo nodo).

Costruire un albero binario

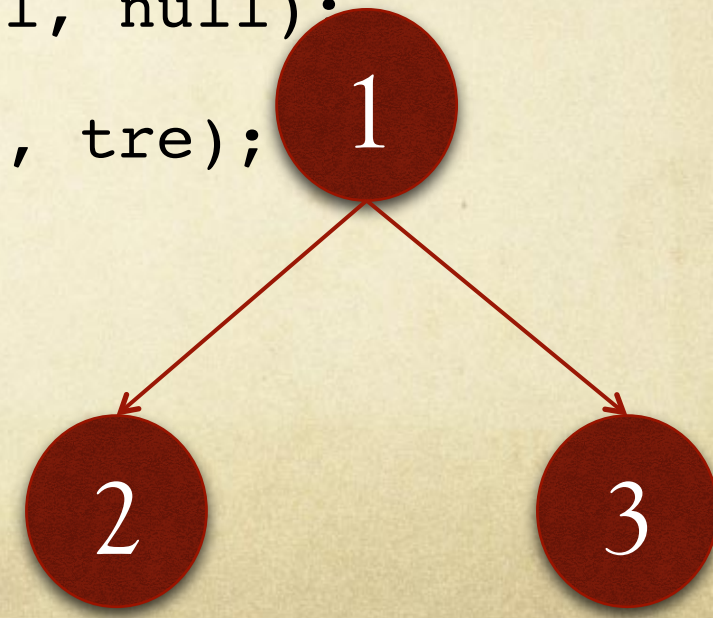
- Creare l'albero come unione di nodi
- Esempio: per creare questo albero:



Costruire un albero binario

- Creare l'albero come unione di nodi
- Esempio: per creare questo albero:

```
Nodo due = new Nodo(2, null, null);

Nodo tre = new Nodo(3, null, null);

Nodo uno = new Nodo(1, due, tre);

Tree t = new Tree(uno);
```



Esercizio

- Implementare il metodo

```
public void insert(Node n)
```

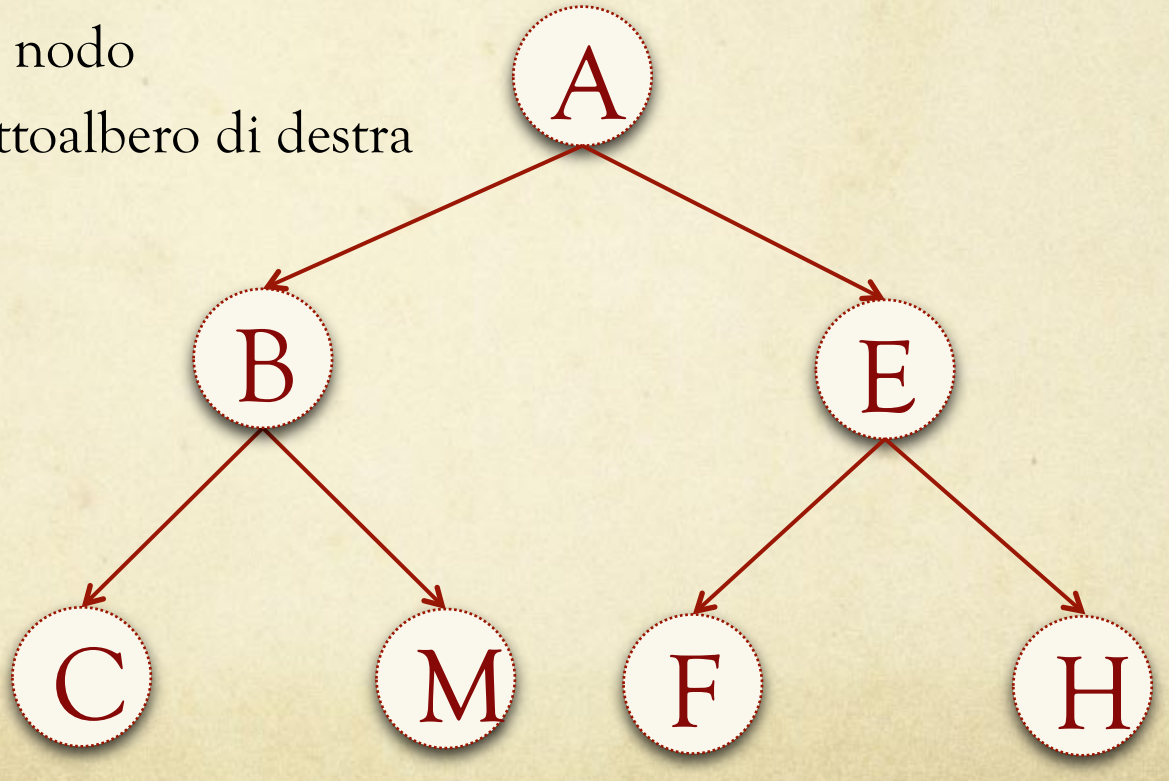
- della classe `Tree` in modo tale che il nodo n venga inserito come foglia dell'albero in posizione random.

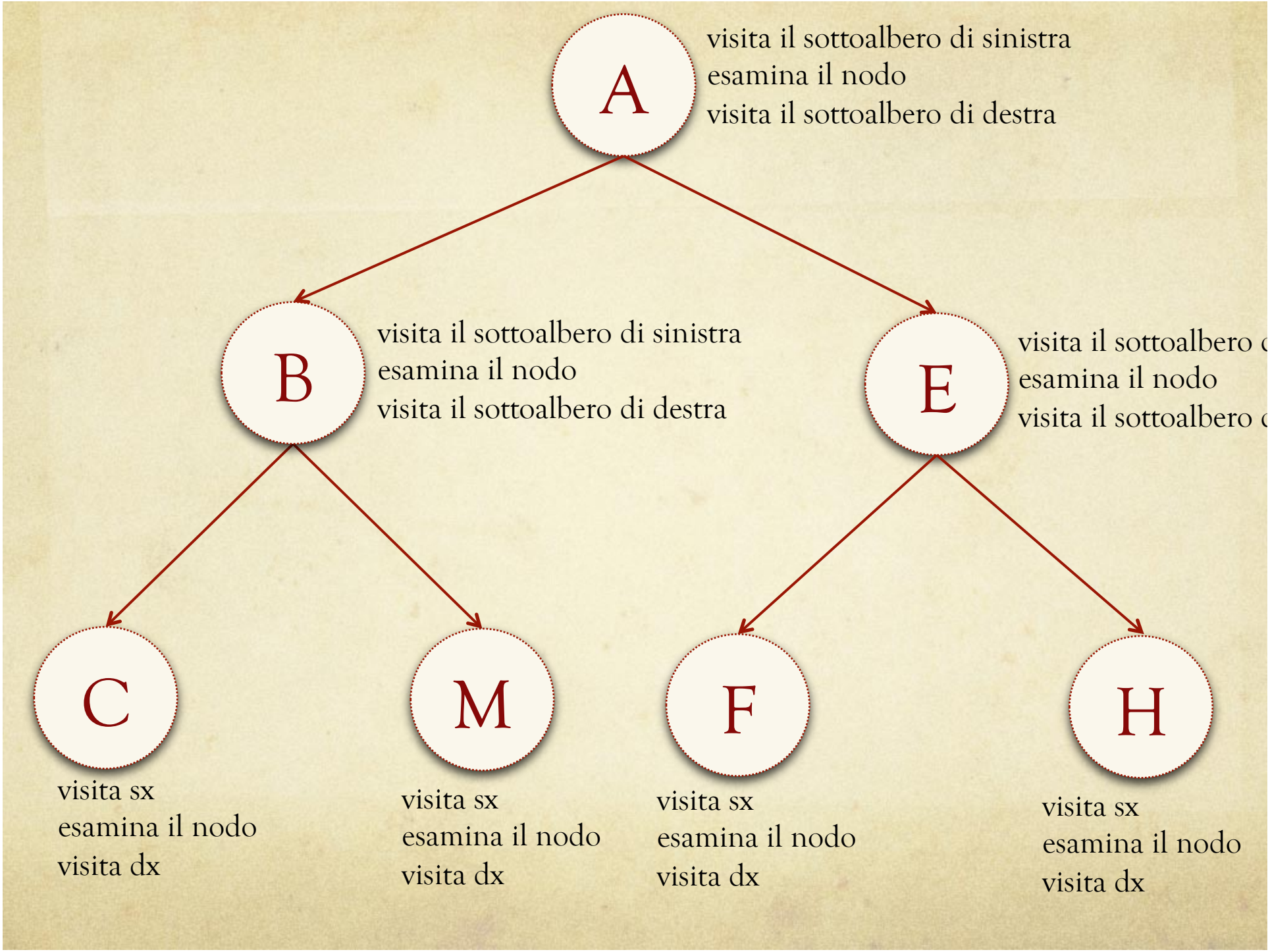
Visite di alberi binari

- Invisita:
  - visita il sottoalbero di sinistra
  - esamina il nodo
  - visita il sottoalbero di destra

Visite di alberi binari

- Invisita:
  - visita il sottoalbero di sinistra
  - esamina il nodo
  - visita il sottoalbero di destra




A
visita il sottoalbero di sinistra
esamina il nodo
visita il sottoalbero di destra
B
visita il sottoalbero di sinistra
esamina il nodo
visita il sottoalbero di destra
E
visita il sottoalbero
esamina il nodo
visita il sottoalbero
C
visita sx
esamina il nodo
visita dx
M
visita sx
esamina il nodo
visita dx
F
visita sx
esamina il nodo
visita dx
H
visita sx
esamina il nodo
visita dx

Implementazione invisita

```
public void inVisit(Node tmp){
   if (tmp != null){
      inVisit(tmp.left);
      System.out.println(" "+tmp.elem);
      inVisit(tmp.right);
   }
}
```

Esercizio

- Modificare il metodo invisita in modo che restituisca una stringa. Tale stringa dovrà contenere il risultato della visita.

```
public String inVisit(Node node){

…

}
```

- Es. nel caso dell'albero visto in precedenza la stringa restituita dalla prima chiamata al metodo invisit conterrà:

  C B M A F E H