Alberi N-ari

alberi binari
: ogni nodo ha 0/1/2 figli

alberi n-ari
: ogni nodo ha un numero qualsiasi di figli

Destro e sinistro?

Negli alberi binari ha senso parlare di "figlio destro" e di "figlio sinistro"

Per gli alberi n-ari no

I sottoalberi di un albero sono un insieme non ordinato

Implementazione di un albero n-ario

Ogni albero è composto da un valore più un insieme non ordinato di alberi

```
import java.util.*;

class NAlbero {
  Object info;
  HashSet figli;
}
```

Albero senza figli

Si può procedere in due modi:

1. il campo `HashSet` vale `null`
2. il campo `HashSet` indica un oggetto che rappresenta l'albero vuoto

Con il secondo sistema, le operazioni sugli alberi risultano semplificate

Quando si crea l'albero, va creato anche l'`HashSet`

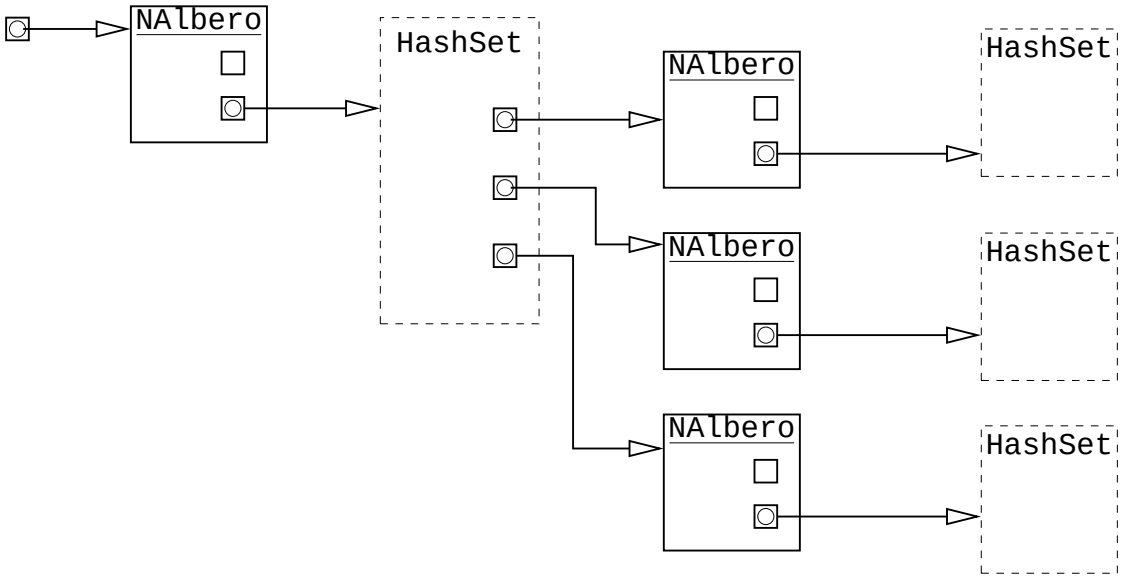
```
import java.util.*;

class NAlbero {
  Object info;
  HashSet figli;

  public NAlbero(Object n) {
    this.info=n;
    this.figli=new HashSet();
  }
}
```

Rappresentazione grafica di un albero

Rappresentazione di un albero con la radice e tre foglie:



Notare che non esiste l'albero vuoto:
se un albero non ha figli, ha un insieme di figli vuoto

Scansione di un albero

Simile a quella degli alberi binari, soltanto che ora c'è un insieme di sottoalberi

```
static void stampaTutti(NAlbero a) {
  if(a==null)
    return;

  System.out.print(a.info+" ");

  Iterator i=a.figli.iterator();
  while(i.hasNext())
    stampaTutti((NAlbero) i.next());
}
```

Usando `null` al posto dell'insieme vuoto:

1. era un errore concettuale (un insieme vuoto di figli non era rappresentato dall'insieme vuoto)
2. prima di fare `a.figli.iterator()` bisognava fare il controllo

Altro esempio: presenza in un albero

Verificare se un albero contiene un intero con un certo valore

```
static boolean presente(NAlbero a, int x) {
  if(((Integer) a.info).intValue()==x)
    return true;

  Iterator i=a.figli.iterator();
  while(i.hasNext())
    if(presente((NAlbero) i.next(), x))
      return true;

  return false;
}
```

Generazione e stampa

Metodi per la generazione di un albero casuale e stampa

```
static NAlbero nalberoRandom(int livelli) {

  int n=(int) (Math.random()*41-20);
  NAlbero a=new NAlbero(new Integer(n));

  if(Math.random()<0.2)
    return a;

  if(livelli<1)
    return a;

  int f=(int) (Math.random()*5);
  for(int i=0; i<f; i++)
    a.figli.add(nalberoRandom(livelli-1));

  return a;
}

static void nalberoStampa(NAlbero a, String before, String after) {
  System.out.print(before.substring(0, before.length()-1));
  System.out.println("+-["+a.info+"]");

  Iterator i=a.figli.iterator();
  while(i.hasNext()) {
    NAlbero f=(NAlbero) i.next();
    if(i.hasNext())
      nalberoStampa(f, before+"  |", before+"  |");
    else
      nalberoStampa(f,  after+"   ",  after+"   ");
  }
}

static void nalberoStampa(NAlbero a) {
  nalberoStampa(a, " ", " ");
}
```

Programma per provare un metodo:

1. genera un albero casuale
2. stampalo
3. invoca il metodo e stampa il risultato

Vedendo l’albero e il risultato del metodo si può capire se il metodo è corretto oppure no