Alberi binari

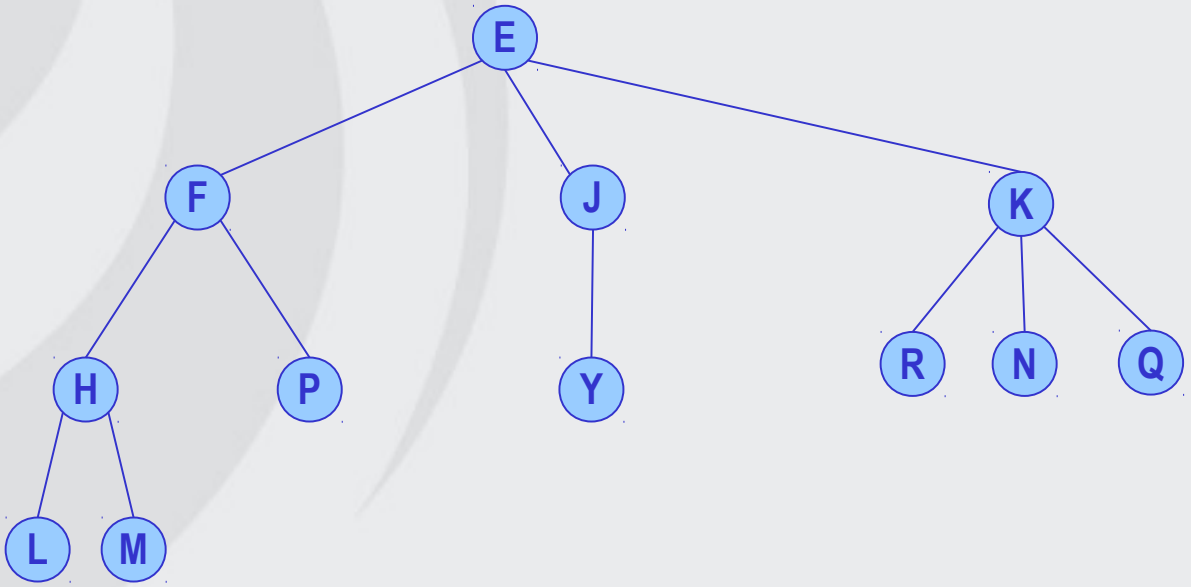
Alberi

Albero binario

Heap tree

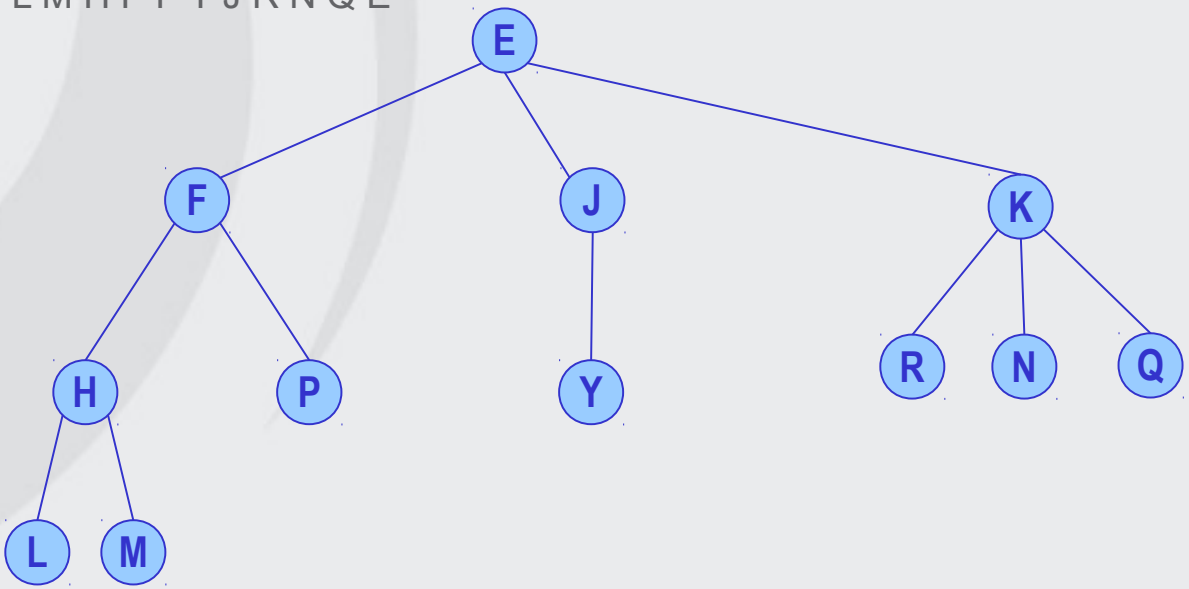
Albero

- Ogni nodo può avere un numero arbitrario di figli
  - Un nodo senza figli viene detto foglia
  - Esiste un nodo particolare chiamato radice, tutti gli altri hanno un solo "padre"



Visita di un albero

- Visita anticipata
  - Si considera il nodo corrente
  - Poi si esaminano uno ad uno i figli se presenti
    - E F H L M P J Y K R N Q
- Visita posticipata
  - Si esaminano i figli
  - Poi si considera il nodo corrente
    - L M H P F Y J R N Q E

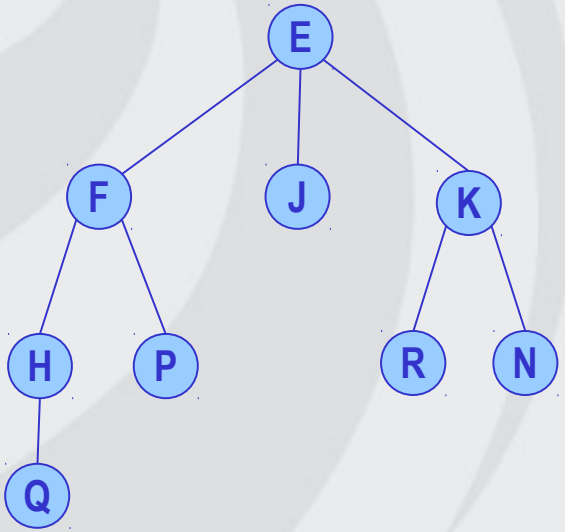


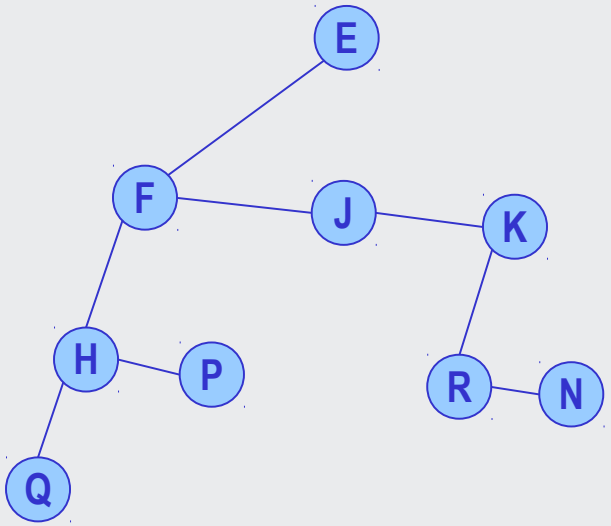
Albero binario

- Ogni nodo ha due figli (destro e sinistro)
- Definizione ricorsiva
  - Caso base: un albero vuoto è un albero binario
  - Caso ricorsivo: un albero è costituito da un nodo (radice) e da due sottoalberi (destro e sinistro)
- Esiste una terza visita per gli alberi binari: la visita simmetrica
  - Si visita il sottoalbero sinistro
  - Si considera la radice
  - Si visita il sottoalbero destro

Trasformazione di alberi

- Da un albero A di N nodi è possibile ottenere un albero binario B di N nodi con il seguente algoritmo:
  - La radice di A coincide con la radice di B
  - Ogni nodo b di B ha come radice del sottoalbero sinistro il primo figlio di b e come sottoalbero destro il fratello successivo di b in A





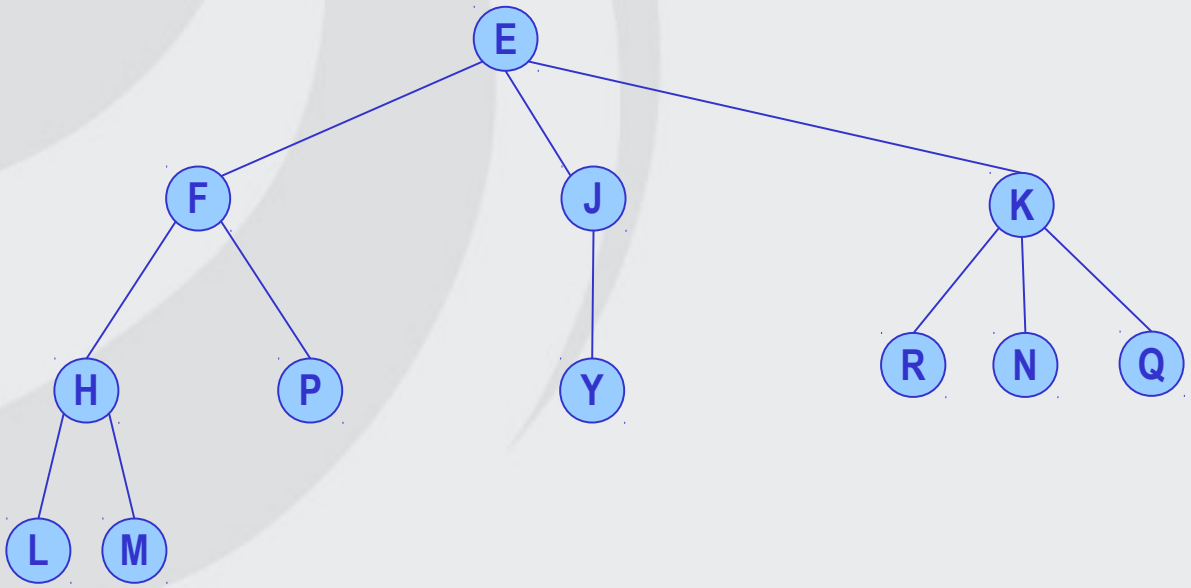
  - La visita posticipata di A coincide con la visita simmetrice di B

Albero binario

- Un albero binario spesso è usato per gestire dati ordinati
- L'albero viene costruito in modo tale che
  - Ogni nodo del sottoalbero sinistro ha valori minori o uguali al valore della radice
  - Ogni nodo del sottoalbero destro ha valori maggiori o uguali al valore della radice
- La visita simmetrica produce i dati ordinati

Ancora visita di un albero

- Visita in ampiezza (per livello)
  - Si considera la radice
  - Si considerano tutti i nodi di livello 1
  - Si considerano tutti i nodi di livello 2
  - ...
  - Si considerano tutti i nodi di livello N
    - E F J K H P Y R N Q L M



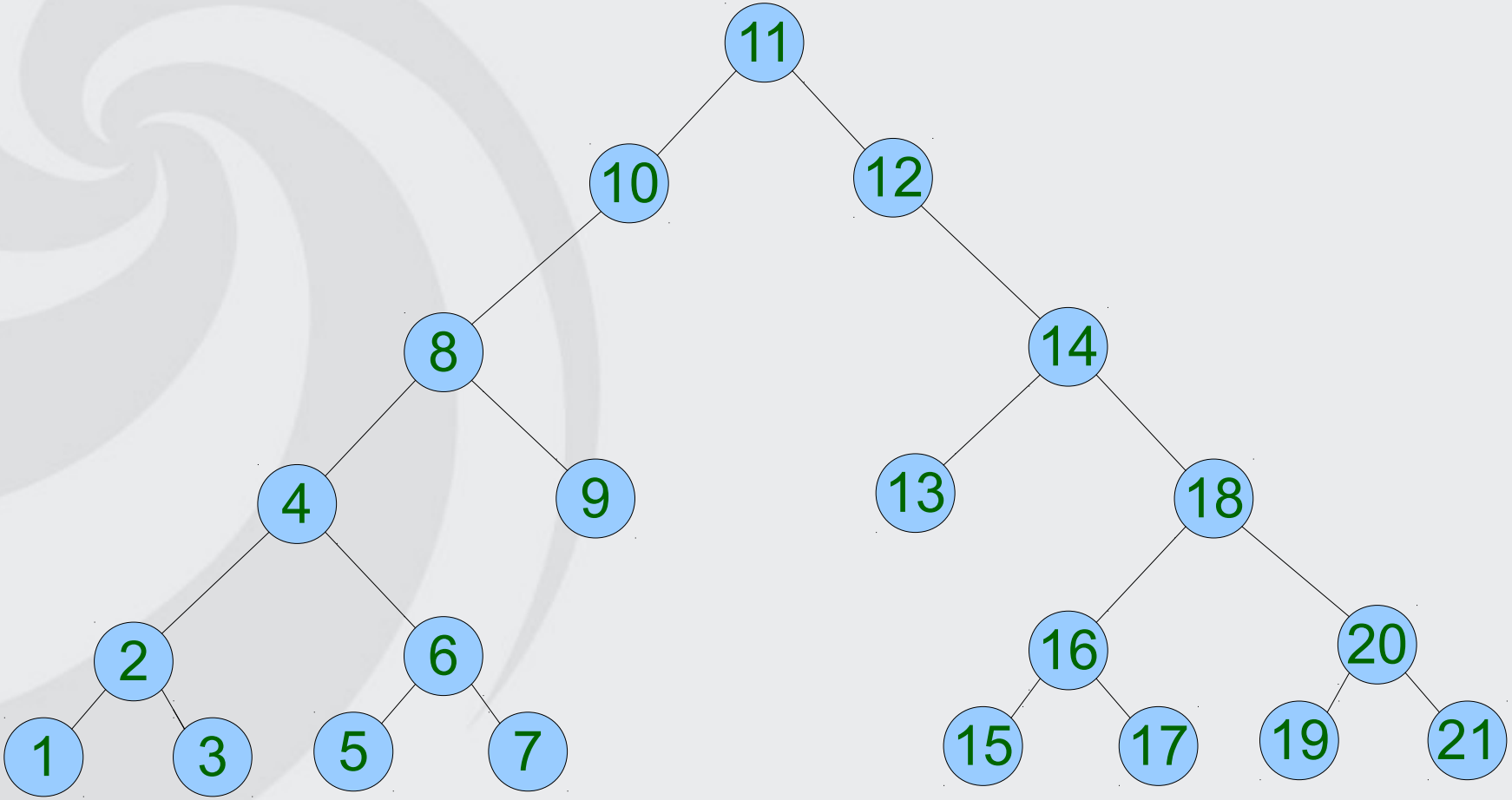
Livello 0

Livello 1

Livello 2

Livello 3

Albero binario



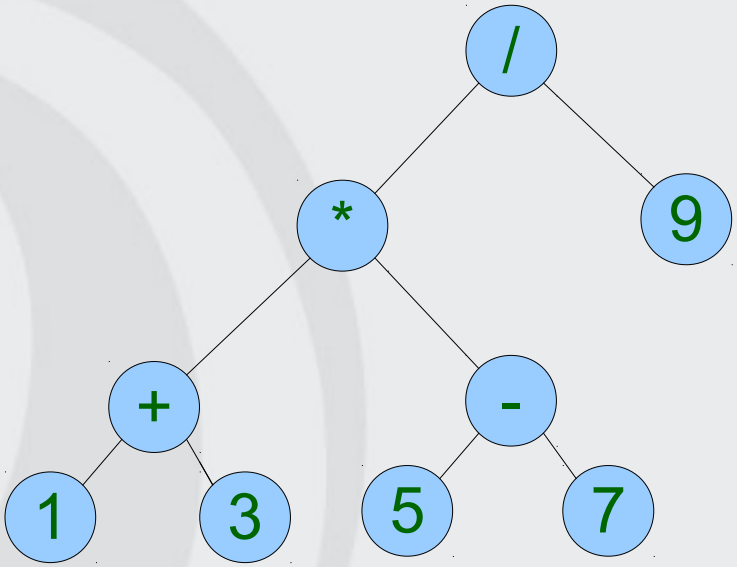
Albero binario

bnode.c
bnode.h
main.c
Makefile

user@ubuntu: ~

```
$ make test
gcc -Wall -I. -c bnode.c
gcc -Wall -I. -o btree bnode.o main.o
./btree<<< "11 10 12 8 14 4 9 13 18 2 6 16 20 1 3 5 7 15 17 19 21"
Visita anticipata
11 10 8 4 2 1 3 6 5 7 9 12 14 13 18 16 15 17 20 19 21
Visita simmetrica
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Visita posticipata
1 3 2 5 7 6 4 9 8 10 13 15 17 16 19 21 20 18 14 12 11
Visita per livello
11 10 12 8 14 4 9 13 18 2 6 16 20 1 3 5 7 15 17 19 21
Livello massimo: 6, profondità minima: 4
```

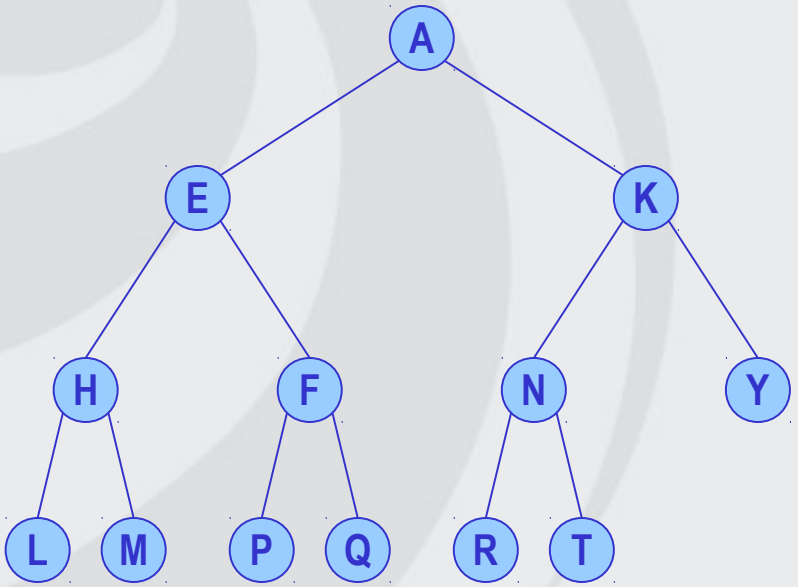
Albero binario e notazioni polacche



Visita anticipata: /*+13-579

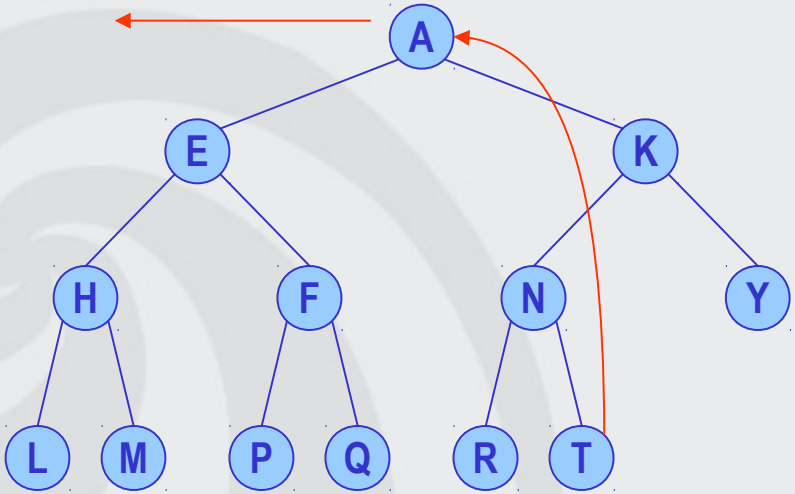
Visita posticipata: 13+57-*9/

Heap tree

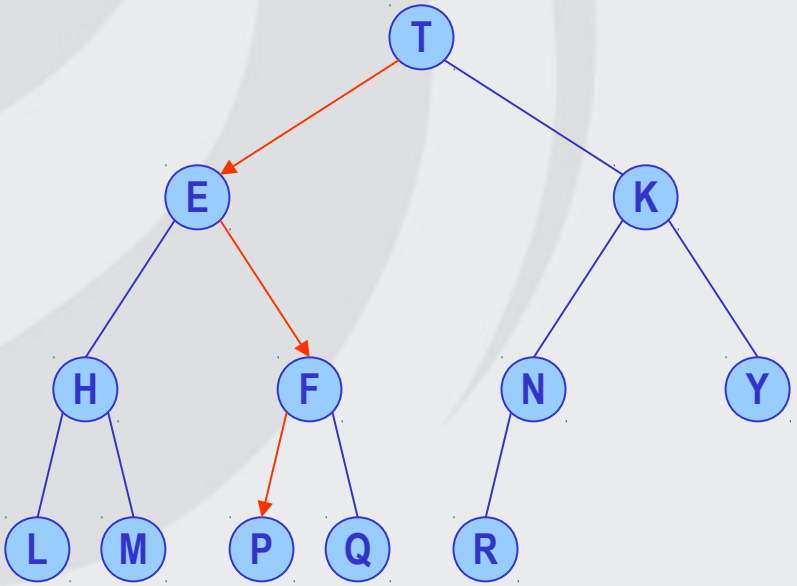


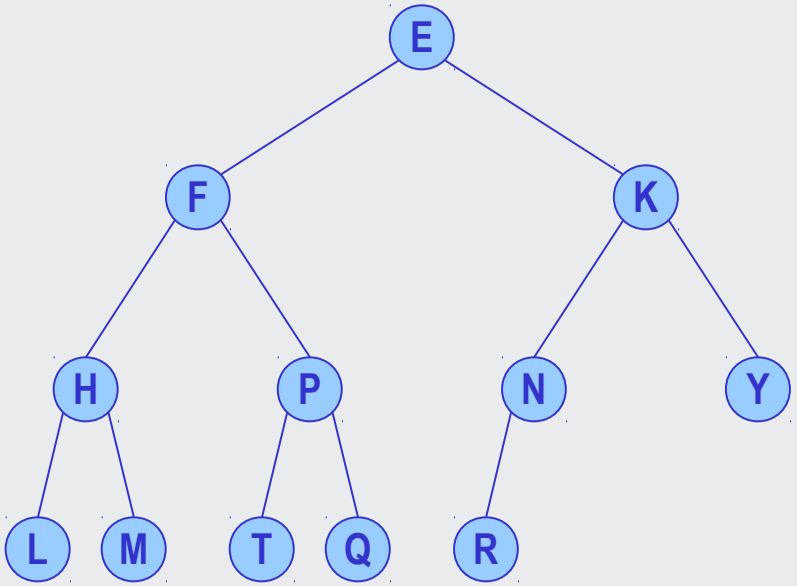
- Proprietà
  - Ogni nodo è maggiore (minore) di tutti gli elementi del sottoalbero di cui è radice
  - L'estrazione avviene alla radice
  - L'inserimento sul primo nodo libero a partire da sinistra dell'ultimo livello
  - L'aggiornamento avviene in un tempo proporzionale al logaritmo della dimensione

Estrazione

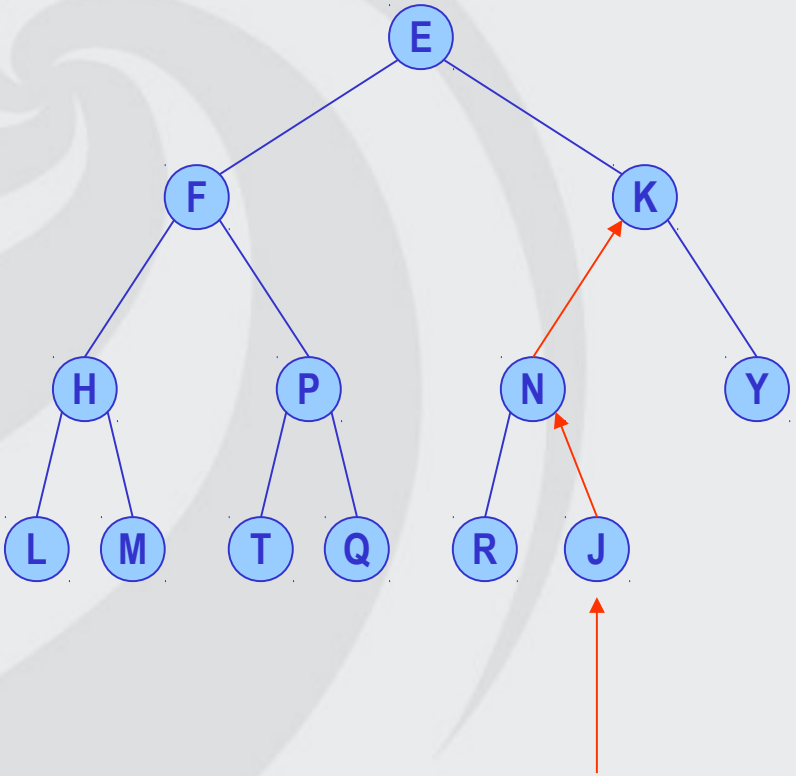


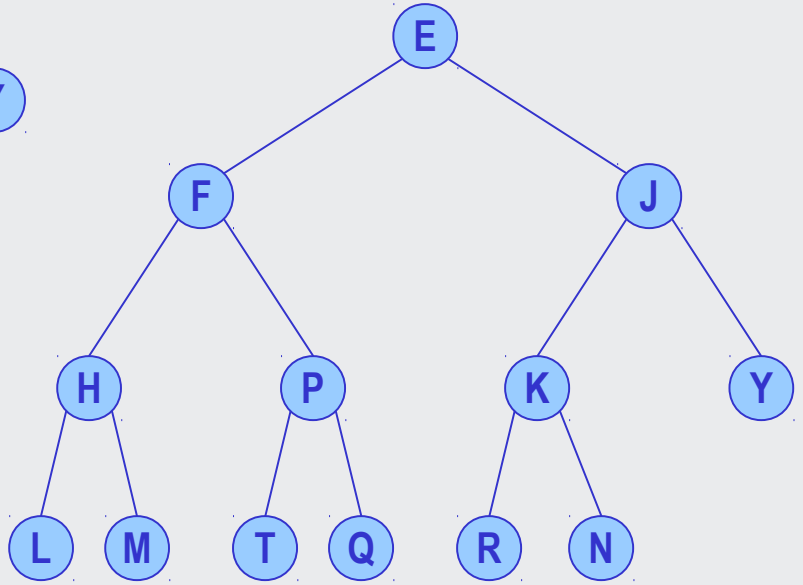
Il nodo più a destra dell'ultimo livello sostituisce la radice, quindi si aggiorna l'albero





Inserimento





Implementazione

- Può essere implementato tramite un vettore
  - La radice è l'elemento di indice 1
  - I figli di un nodo di indice i sono memorizzati nelle posizioni (2*i) e (2*i+1)
- Un albero di dimensione N occupa gli elementi del vettore da 1 a N
  - L'estrazione coinvolge l'indice 1
  - L'inserimento l'indice N+1
  - Non rimangono sono mai *buchi* nel vettore
- L'albero è riempito su tutti i livelli tranne eventualmente l'ultimo
  - Non rimangono mai *buchi* nel vettore
  - L'albero è bilanciato

Heap tree

heap.c

user@ubuntu: ~

```
$ for((i=0; i<15; i++)) do echo $((RANDOM % 100));done | ./heap
77.000000 60.000000 71.000000 52.000000 57.000000 33.000000
62.000000  8.000000 44.000000 19.000000 31.000000 10.000000
17.000000 26.000000 33.000000
77.000000 71.000000 62.000000 60.000000 57.000000 52.000000
44.000000 33.000000 33.000000 31.000000 26.000000 19.000000
17.000000 10.000000  8.000000
```

Produce 15 interi fra 0 e 15

Il risultato è una stampa dei valori in ordine decrescente