# Strutture Dati

## Lezione 10
## Gli alberi

## Oggi parleremo di …

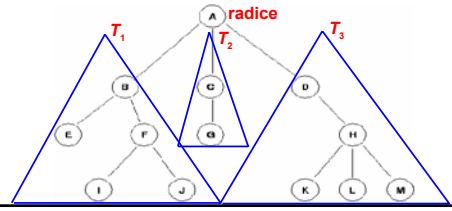
- Gli alberi
  - definizione
  - elementi di un albero
  - rappresentazione
- Alberi binari
  - ADT
  - proprietà
  - rappresentazione
    - mediante array
    - mediante collegamenti

## Gli alberi

- Gli **alberi** sono strutture dati in cui i dati sono correlati attraverso rami
  - genealogia
  - pedigree
  - lingue

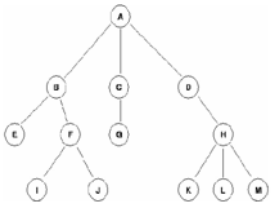

## Gli alberi: definizione

- Si definisce **albero** una struttura dati costituita da un insieme finito di uno o più nodi dove:
  - esiste un nodo speciale chiamato radice (root);
  - i nodi restanti sono suddivisi in $n$ ($n \geq 0$) insiemi disgiunti $T_1$, $T_2$, …, $T_n$, dove ciascuno di questi sottoinsiemi è una struttura ad albero, detta sottoalbero.



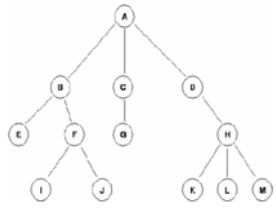

## Gli alberi: elementi

- **Nodo**: i nodi sono costituiti sia dalle informazioni che dai rami che si collegano ad altri nodi.
- **Grado di un nodo**: si definisce grado di un nodo il numero di rami del nodo.
- **Grado di un albero**: è il massimo valore dei gradi di ciascun suo nodo.
- **Foglia**: sono i nodi aventi grado nullo (sono detti anche nodi terminali).
- **Padre**: un nodo di grado non nullo è padre delle radici dei sottoalberi.
- **Fratelli**: sono i nodi aventi lo stesso padre.



## Gli alberi: elementi

- **Antenati di un nodo**: sono i nodi che si trovano nel percorso dalla radice al nodo.
- **Discendenti di un nodo**: sono i nodi che si trovano nei sottoalberi del nodo.
- **Livello di un nodo**: per ogni nodo è pari al livello del padre aumentato di 1; il nodo radice ha livello 1.
- **Altezza** (o **profondità**) **di un albero**: è il livello massimo di un nodo dell'albero.



**La profondità è 4**

# Gli alberi: rappresentazione

- La rappresentazione degli alberi si può ottenere mediante le rappresentazioni collegate (con liste)
  - il grado di un nodo può essere anche molto elevato e comunque non noto a priori;
  - ogni nodo avrà dimensione variabile.

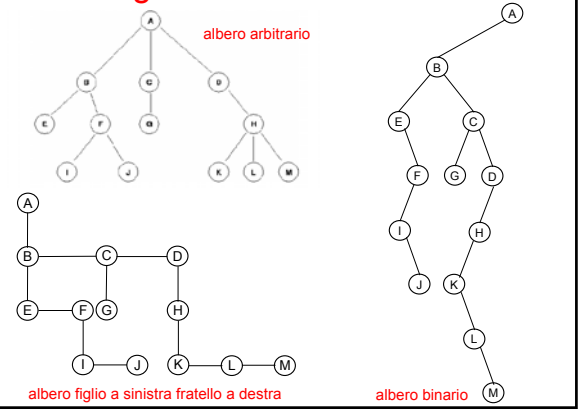
| dati | link1 | link2 | ..... | linkN |
|---|---|---|---|---|

- La rappresentazione immediata degli alberi porta a nodi di dimensioni non costanti.
- Una possibile alternativa è quella detta figlio a sinistra fratello a destra
  - si possono utilizzare nodi di dimensione predeterminata;
  - permette di rappresentare un generico albero di grado *n* con un albero di grado 2.

# Gli alberi figlio a sinistra fratello a destra

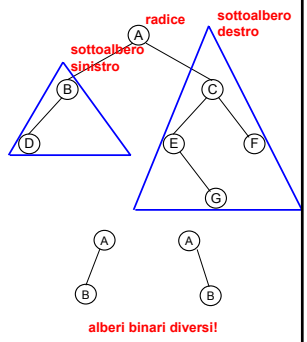
- La rappresentazione figlio a sinistra, fratello a destra permette sempre di ricondurre un albero generico in un albero di grado due.
- Gli alberi di grado due sono detti alberi binari.
- Per ottenere una rappresentazione di un albero come albero di grado 2, basta ruotare di 45 gradi in senso orario la rappresentazione figlio a sinistra fratello a destra.

# Gli alberi figlio a sinistra fratello a destra




# Gli alberi binari

- Si definisce **albero binario** un insieme finito di nodi che può essere anche vuoto formato da un nodo radice e da due alberi binari disgiunti detti **sottoalbero sinistro** e **sottoalbero destro**
  - gli alberi binari possono essere anche vuoti;
  - vi è distinzione fra i nodi;
  - vale la stessa terminologia degli alberi.




# Il tipo di dati astratto *Albero_binario*

```
Struttura AlBin
   oggetti: un insieme di finito di nodi sia vuoto sia formato da un
        nodo radice, da un AlBin sinistro e da un AlBin destro.

   funzioni: per ogni ab, ab1, ab2 ∈ AlBin, item ∈ elemento

    AlBin Create()              ::= crea un albero binario vuoto
    Booleano IsEmpty(ab)        ::= if(ab == albero binario vuoto)
                                    return TRUE else return FALSE.
    AlBin CreaAB(ab1, item, ab2) ::= return un albero binario il cui
                                    sottoalbero sinistro è ab1, il cui
                                    sottoalbero destro è ab2 e il cui
                                    nodo radice contiene il dato item.
    Albin FiglioSx(ab)          ::= if(IsEmpty(ab)) return errore else
                                    return il sottoalbero sinistro di ab.
    elemento Dati(ab)           ::= if(IsEmpty(ab)) return errore else
                                    return i dati nel nodo radice di ab.
    Albin FiglioDx(ab)          ::= if(IsEmpty(ab)) return errore else
                                    return il sottoalbero destro di ab.
end AlBin
```

# Gli alberi binari: proprietà

- Il numero massimo dei nodi di livello *i* di un albero binario è $2^{i-1}$, con $i \geq 1$.
- Si dimostra per induzione su *i*.
  - per la radice ($i=1$) è immediato.
  - in un albero binario ogni nodo può avere al massimo grado 2.
  - per il livello *i* il numero massimo dei nodi è pari a 2 volte il numero massimo dei nodi di livello *i*-1

$$2 \times 2^{i-2} = 2^{i-1}.$$

- Il numero massimo dei nodi di un albero binario di profondità *k* è $2^k-1$, con $k \geq 1$.

$$\sum_{i=1}^{k} 2^{i-1} = 2^k - 1.$$

- Gli alberi binari hanno un numero massimo di nodi sempre dispari.

## Gli alberi binari: proprietà

- Per qualsiasi albero binario non vuoto, se $n_2$ è il numero di nodi di grado 2 e $n_0$ il numero delle foglie si ha che $n_0 = n_2 + 1$.
- Il numero totale di nodi $n$ sarà pari alla somma del numero di nodi di grado 0, 1 e 2, ovvero

$$n = n_0 + n_1 + n_2.$$

- Ogni nodo è collegato al padre tramite un ramo, eccezion fatta per la radice. Detto $B$ il numero dei rami:

$$n = B + 1.$$

- I rami derivano dai nodi di grado non nullo:

$$B = n_1 + 2 \times n_2.$$

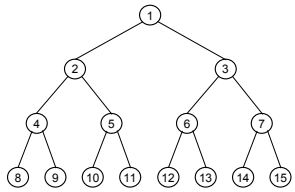
da cui si deduce che

$$n = n_1 + 2 \times n_2 + 1.$$

- Sottraendo quest'ultima relazione dalla prima si ottiene ciò che si voleva dimostrare.

## Gli alberi binari: proprietà

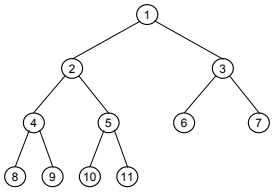
- Un **albero binario** di profondità $k$ si dice **pieno** quando è costituito da $2^k$-1 nodi.




albero binario pieno di profondità 4

## Gli alberi binari: proprietà

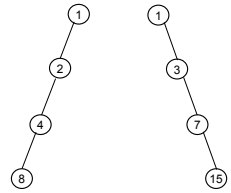
- Un **albero binario** di profondità $k$ e $n$ nodi si dice **completo** se i suoi nodi corrispondono ai nodi da 1 a $n$ dell'albero pieno.




albero binario completo di profondità 4

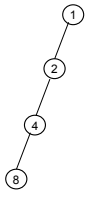
## Gli alberi binari: proprietà

- Un **albero binario** si dice **sbilanciato a sinistra** se è inclinato sulla sinistra, ovvero se ogni nodo è figlio a sinistra di suo padre.
- Analogamente si definisce **albero binario sbilanciato a destra**.




## Rappresentazione degli alberi binari

- Una possibile rappresentazione degli alberi si può ottenere **mediante gli array**.
- In questo caso è possibile risalire ad ogni nodo mediante la sua posizione nell'array.
- Tale rappresentazione è possibile per tutti gli alberi ma ci sarà spazio sprecato.




| |
|---|
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |

## Rappresentazione degli alberi binari

- Dato un albero binario completo con $n$ nodi per un nodo di indice $i$ con $1 \leq i \leq n$, valgono le seguenti proprietà:
  - il padre del nodo di indice $i$ si trova nella posizione $\lfloor i/2 \rfloor$, se $i = 1$ è la radice.
  - il figlio sinistro del nodo di indice $i$ si trova in $2i$ se $2i \leq n$ altrimenti non vi è un figlio sinistro;
  - il figlio destro si trova nella posizione $2i + 1$ se $2i + 1 \leq n$ altrimenti non vi è un figlio destro.
- Il 1° punto discende dal 2° mentre il 2° e il 3° si dimostrano per induzione.
- **Dim. 2° punto**. Per $i = 1$, il figlio sinistro si trova in posizione 2 .
- **Hp**. Per ogni $j$, $1 \leq j \leq i$, figlio sinistro di $j$ si trova in $2j$.
- I nodi che precedono figlio sinistro di $i+1$ sono i figli a destra e a sinistra di $i$.
- Il figlio sinistro di $i$ si trova in $2i$, quindi il figlio a sinistra di $i+1$ si trova in $2i+2 = 2(i+1)$.

# Rappresentazione degli alberi binari

- La rappresentazione mediante array soffre delle generiche inadeguatezze delle rappresentazioni sequenziali.
- E' preferibile una rappresentazione collegata:

```
struct nodo {
        miodato  dato;
        struct nodo     *sinistra;
        struct nodo     *destra;
        };
struct nodo *mytree;
```

- Al pari delle liste singolarmente collegate NON è semplice determinare il padre, comunque è possibile aggiungere un puntatore per fare questo.

# Rappresentazione degli alberi binari