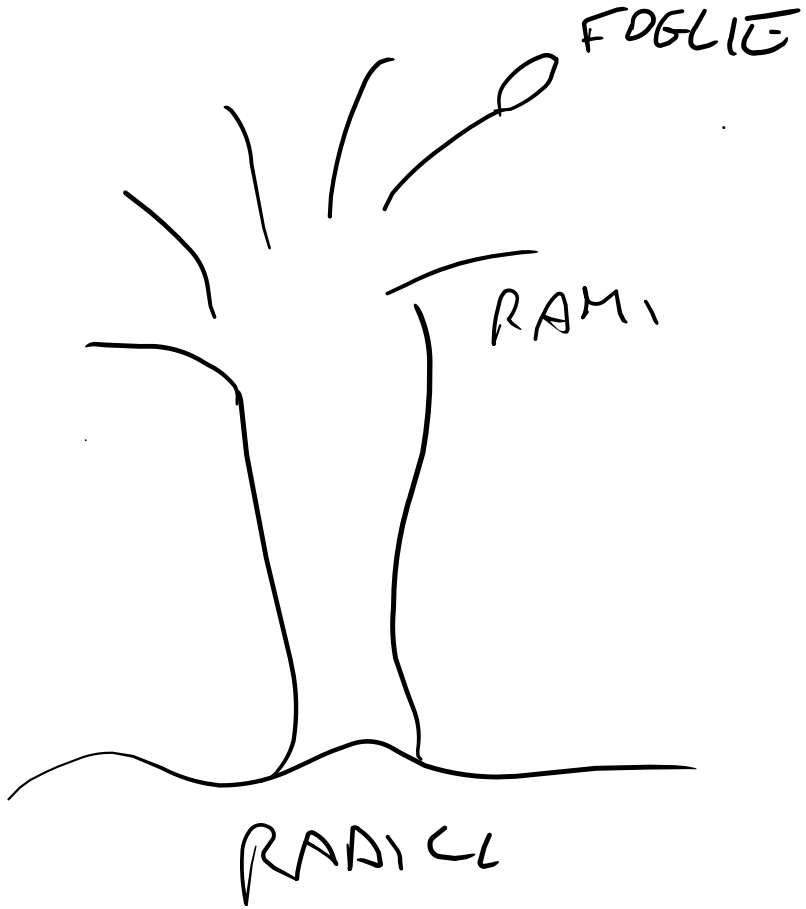
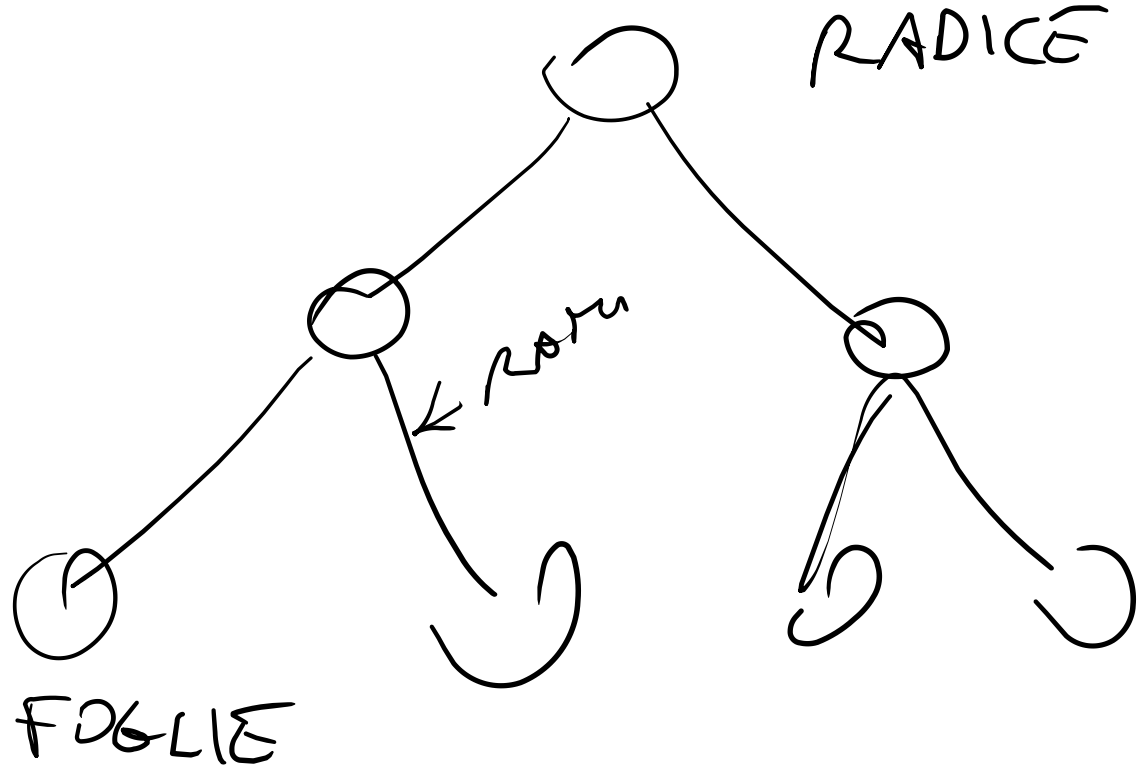
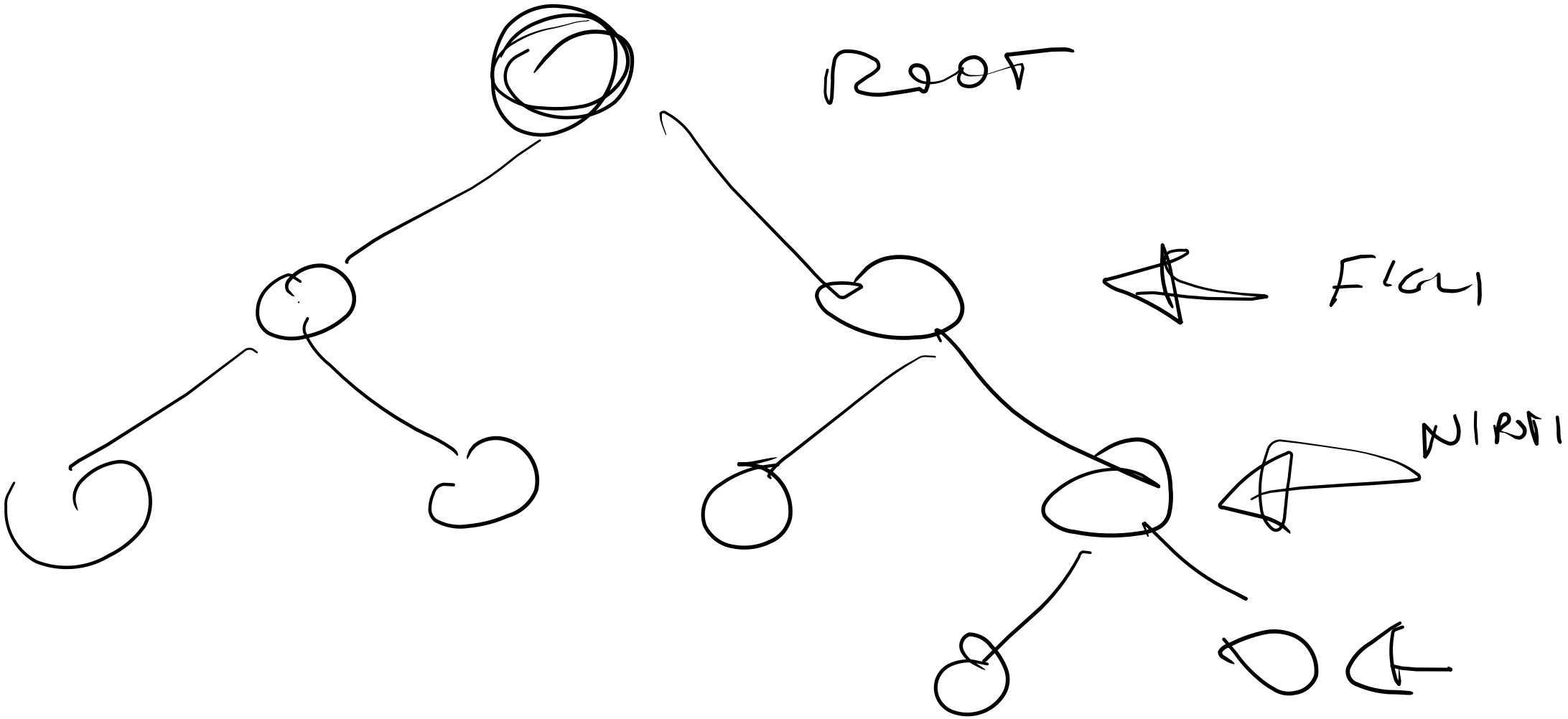
alberi binari di ricerca

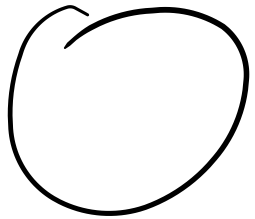


NATURA



INFORMATICA


ROOT
FIGU
NIRTI

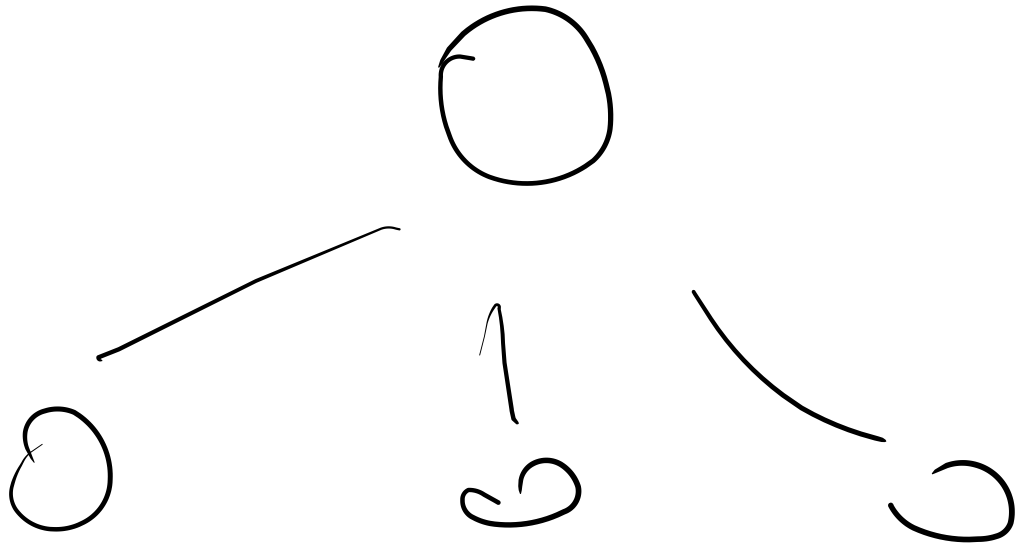


NODO

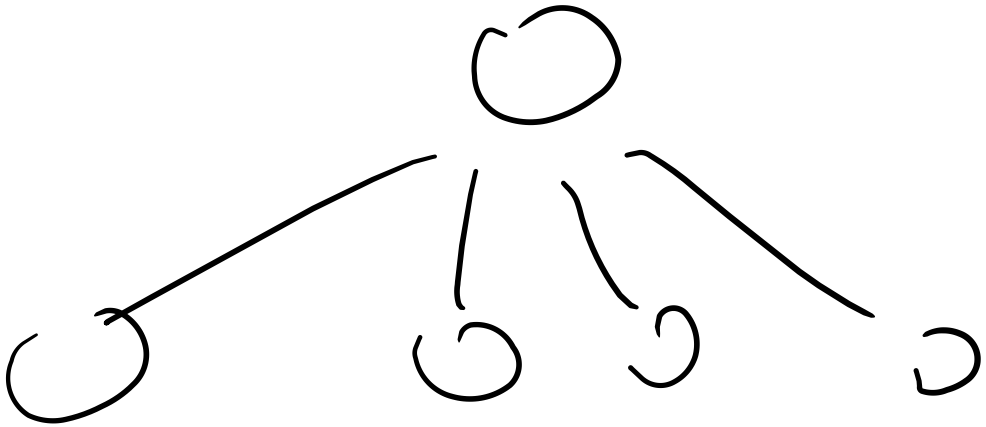
ARIETÀ = NUMERO DI FIGLI POSSIBILI

FIGLI → O...O ELENCO DI NODI

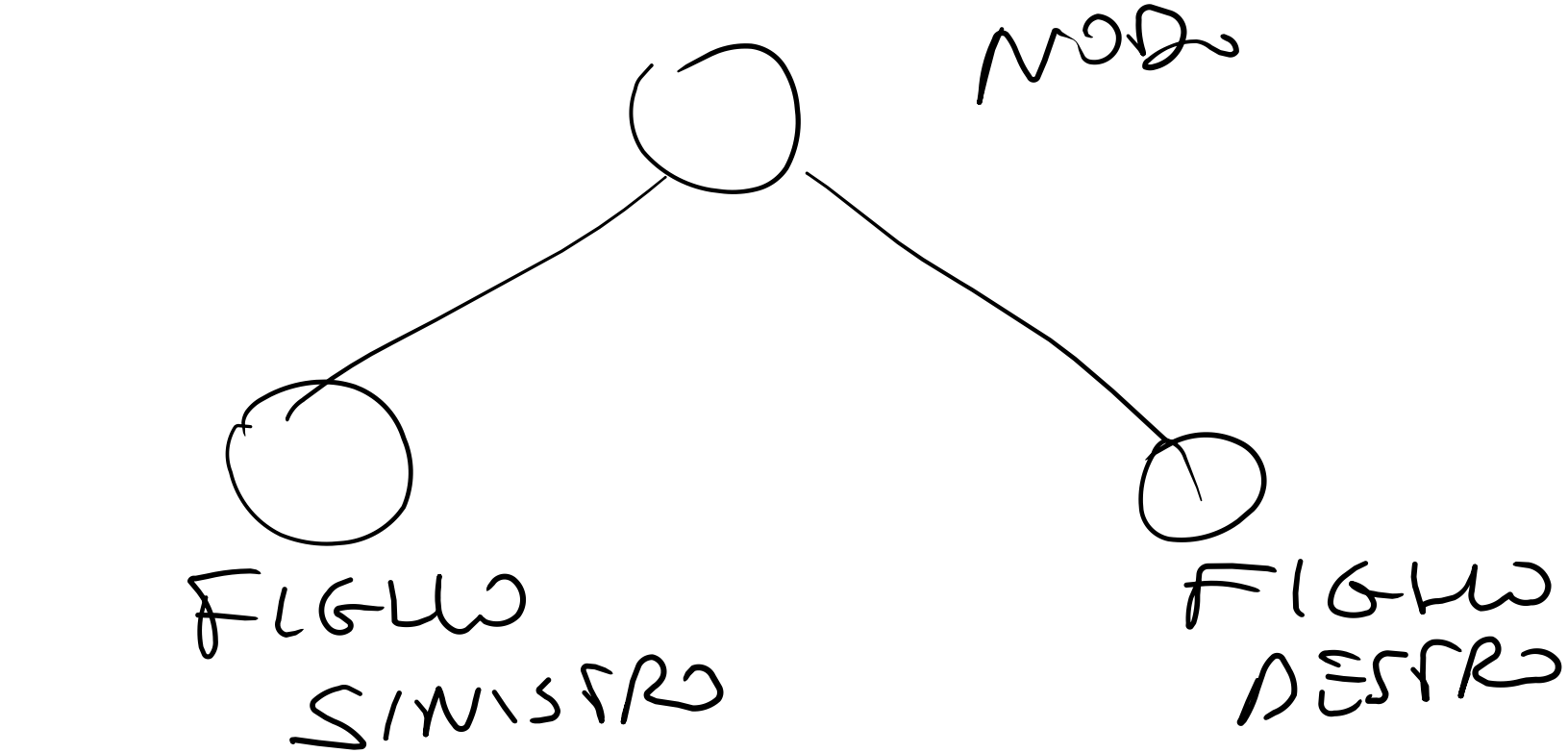
PARENT → O NODO



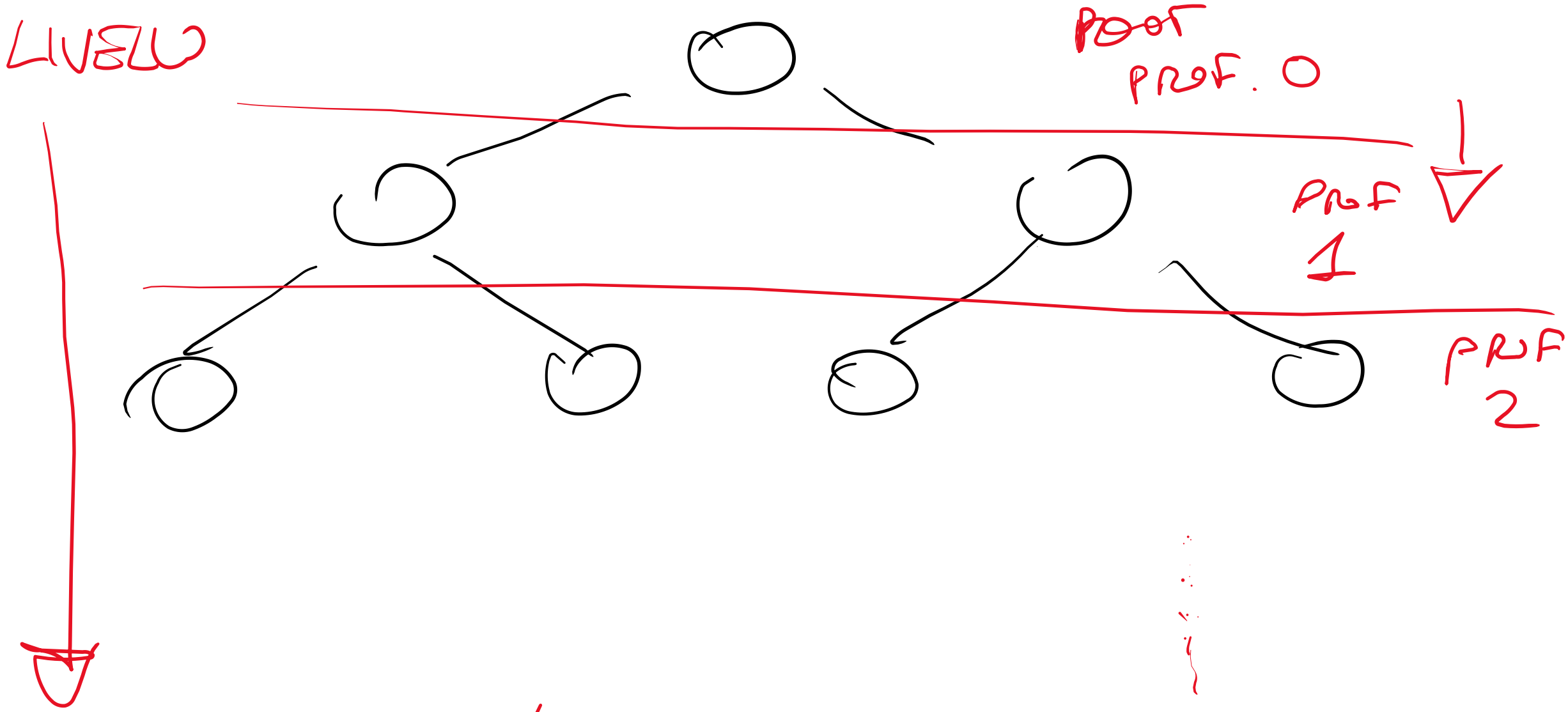
TERNARIO



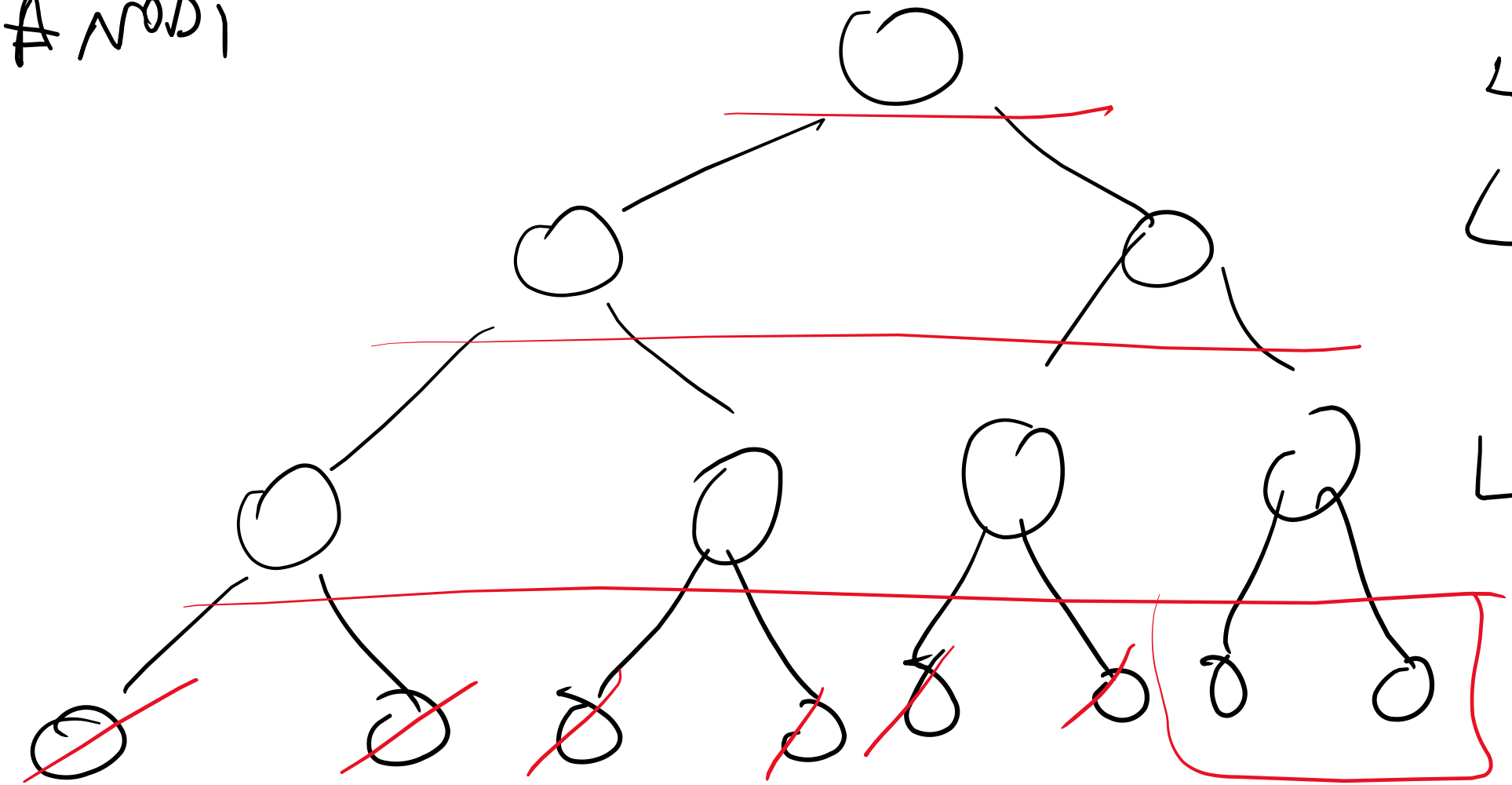
QUATERNARIO



BINARIO


LIVELLO
ROOT
PROF. 0
PROF
1
PROF
2
ALTEZZA DELL'ALBERO = # DI LIVELLI

NODI



LIV 0 : 2^0

LIV 1 : 2^1

LIV 2 : 4^{2^2}

LIV 3 : 8^{2^3}

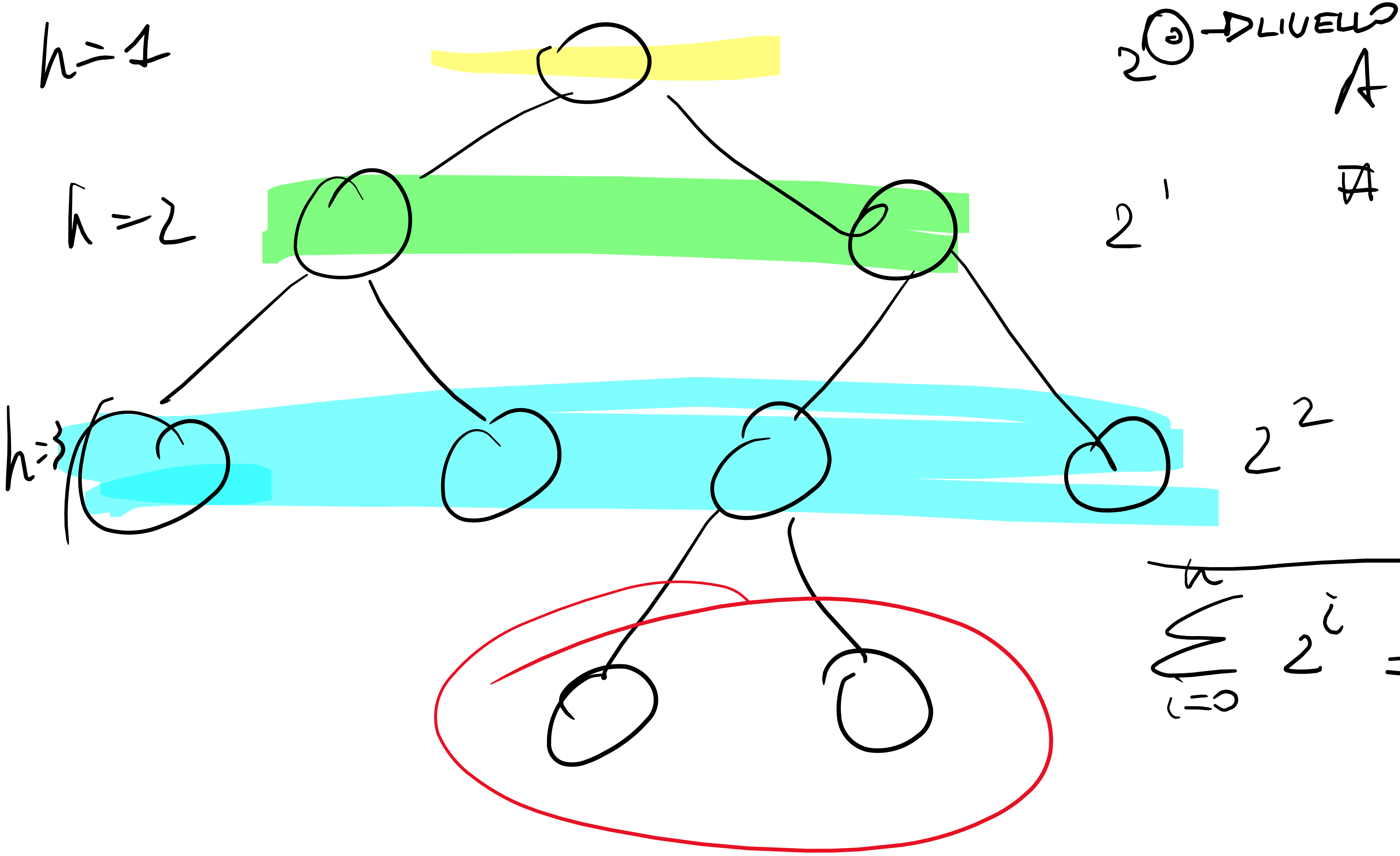
LIV 4 : 16^{2^4}

nuovi di nodi $= 2^h - 1 \rightarrow$

nodi $\leq 2^{h-1} \rightarrow n \leq 2^{h-1} - 1 \rightarrow$

$h = \lceil \log_2 n \rceil$

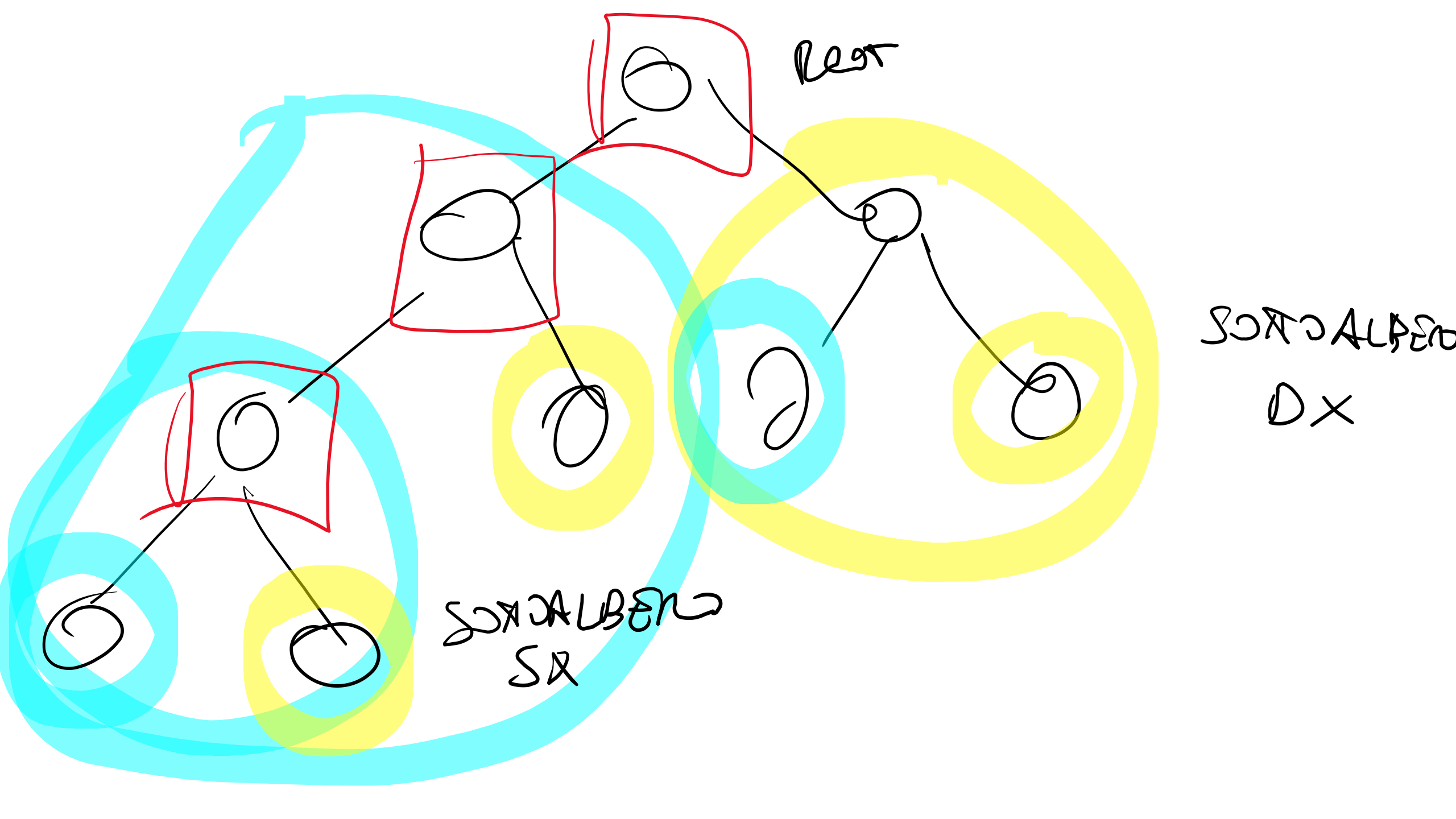
UN ALBERO BINARIO È COMPLETO SE E SOLO SE HA ESATTAMENTE 2^h NODI, DOVE h È L'ALTEZZA DELL'ALBERO



Altezze : 4

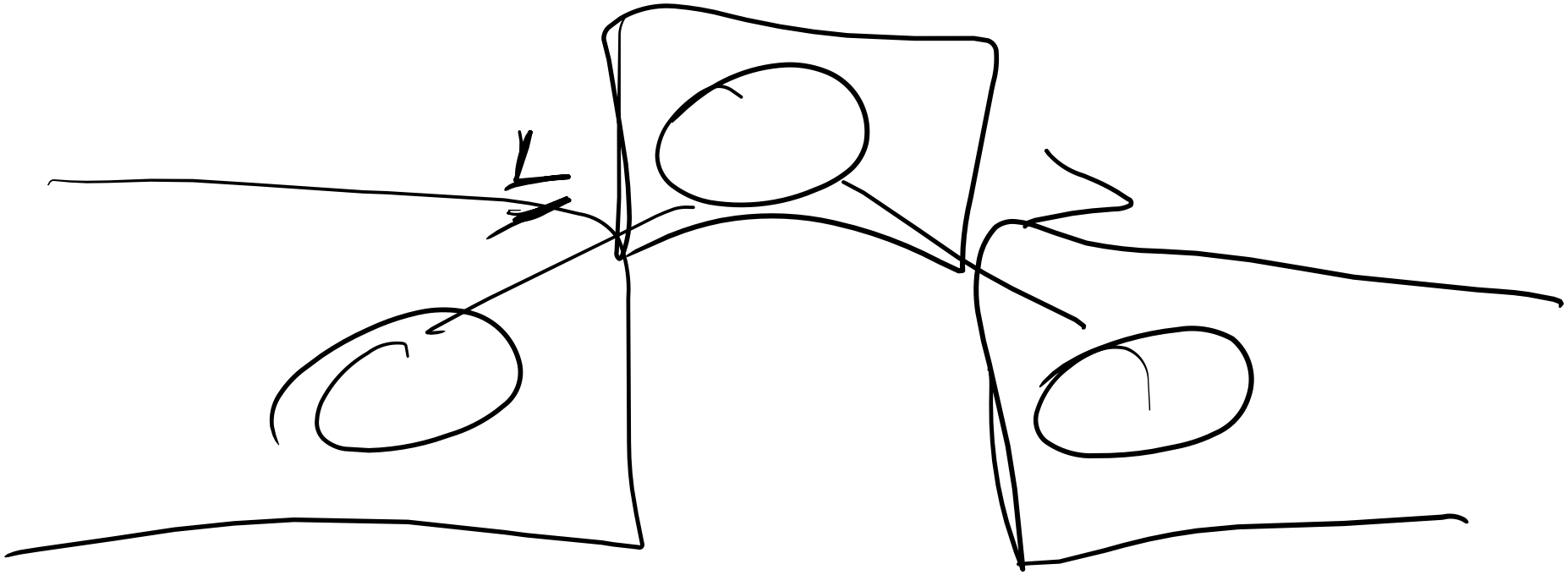
nodi : 15

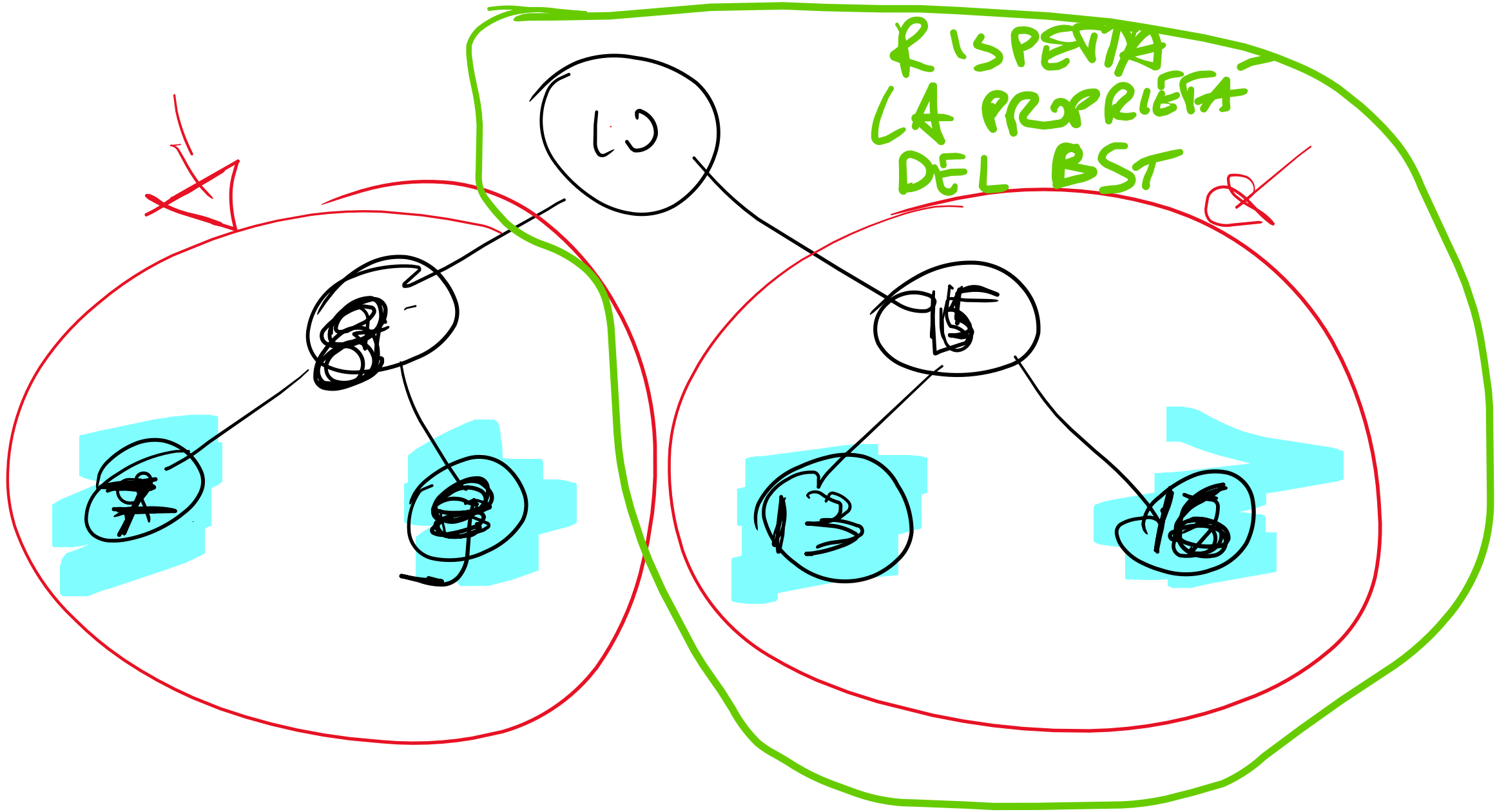
$$\sum_{i=0}^{h} 2^i = 2^h - 1$$

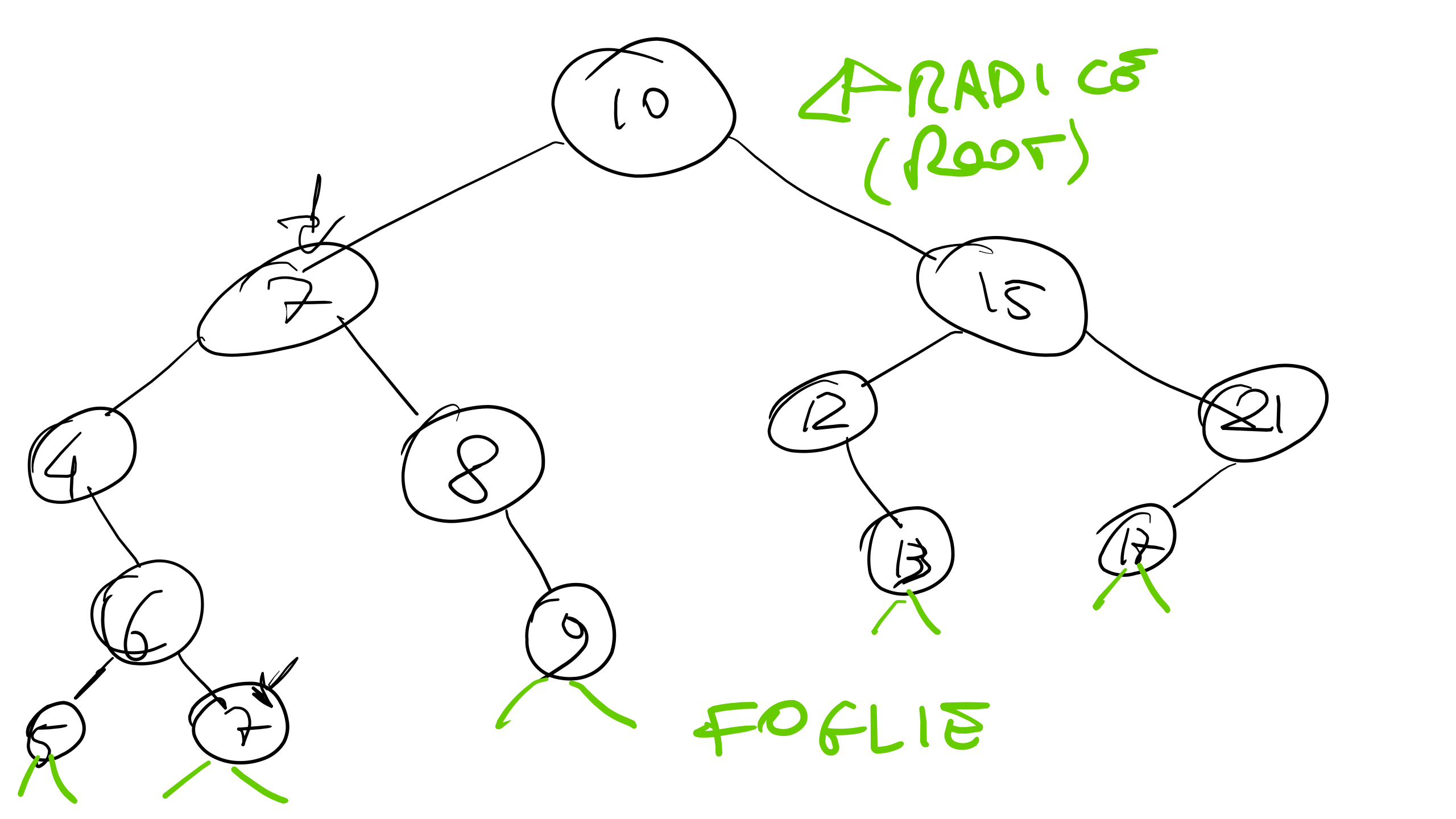

Root
SOTTOALBERO
DX
SOTTOALBERO
SX

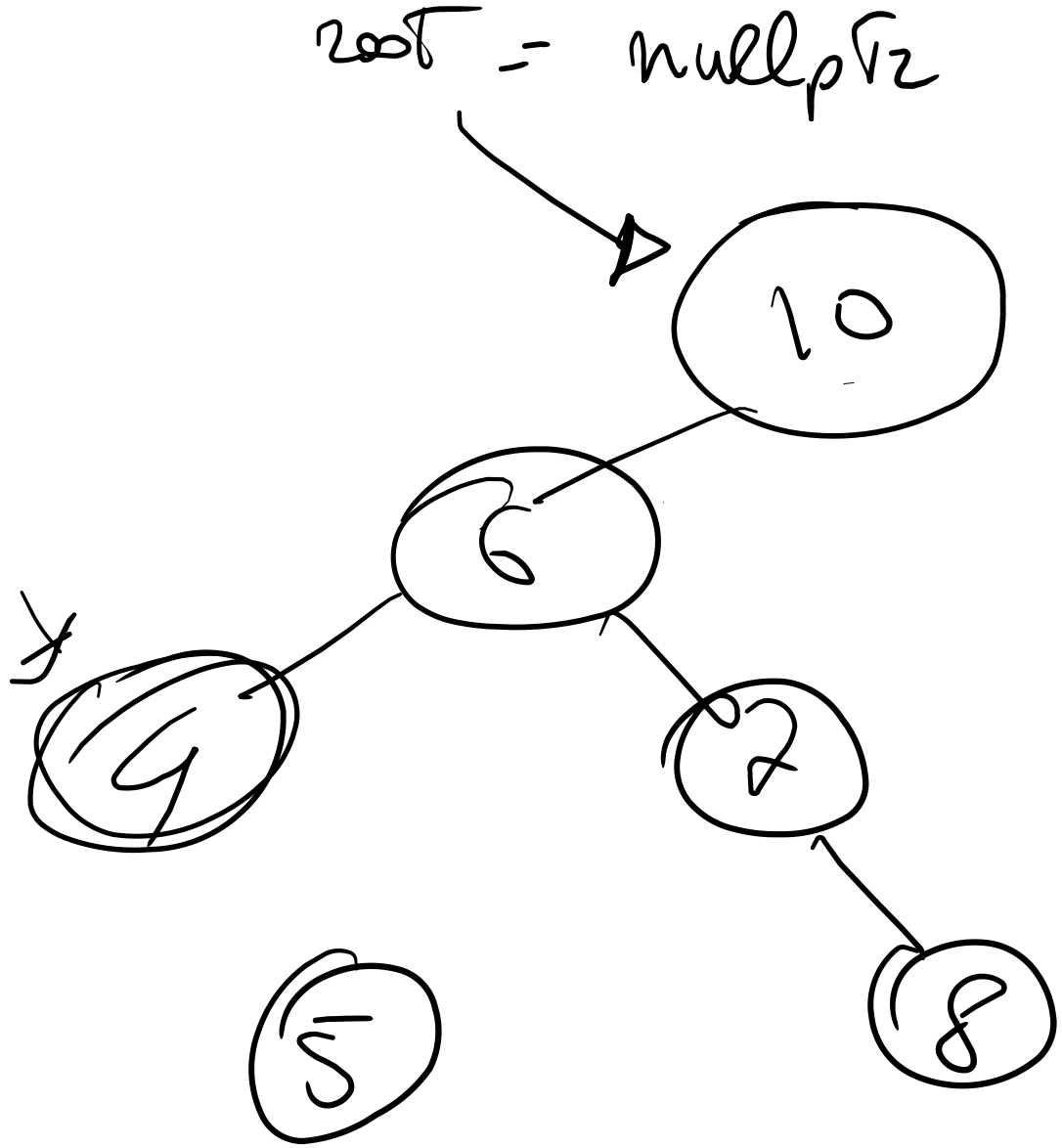
ALBERO BINARIO DI RICERCA

(BST - Binary Search Tree)




RISPETTA
LA PROPRIETÀ
DEL BST


10
RADICE
(ROOT)
7
15
4
8
12
21
6
9
13
17
5
7
FOGLIE



insert (10)

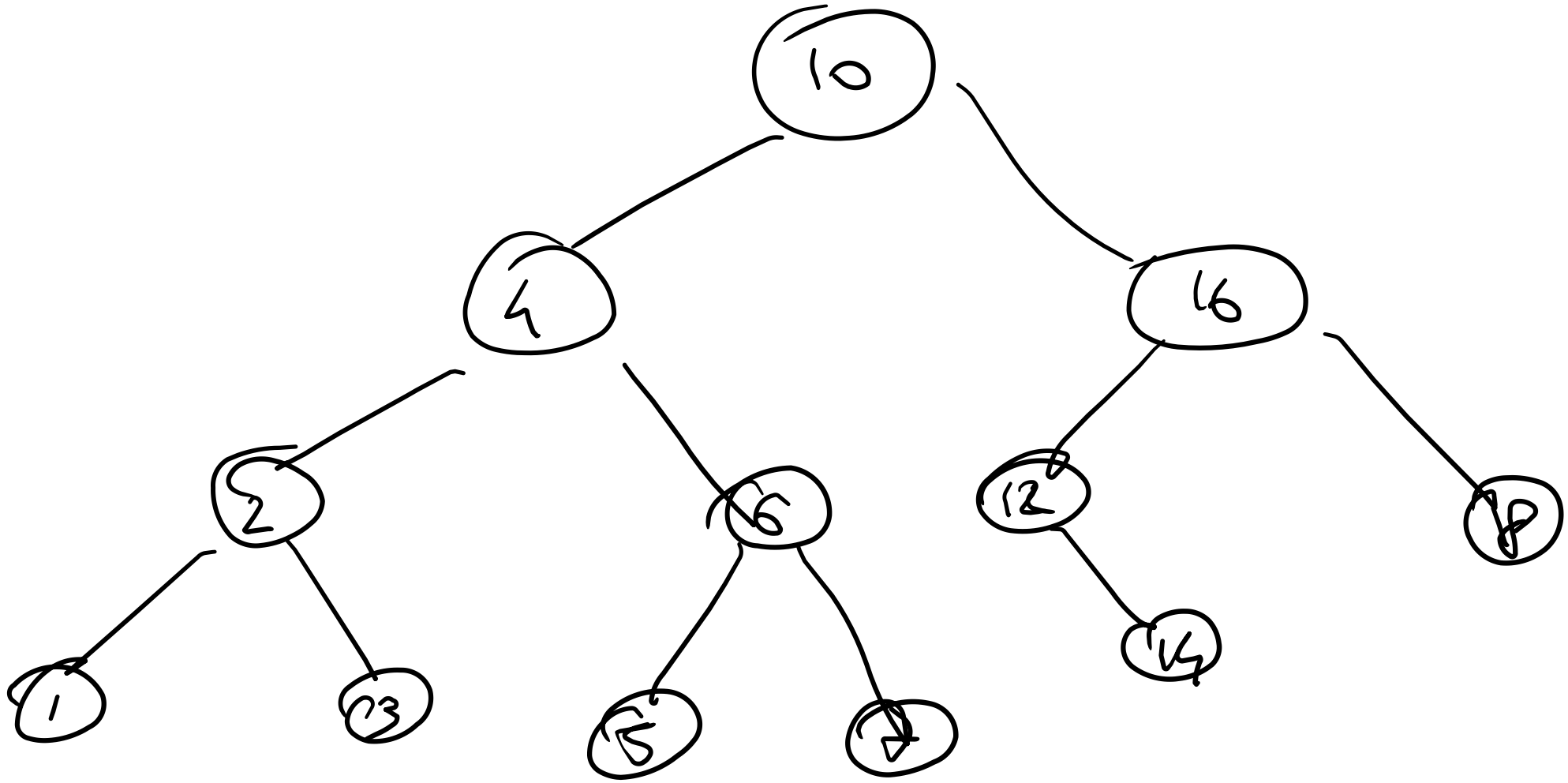
insert (6)

insert (4)

insert (2)

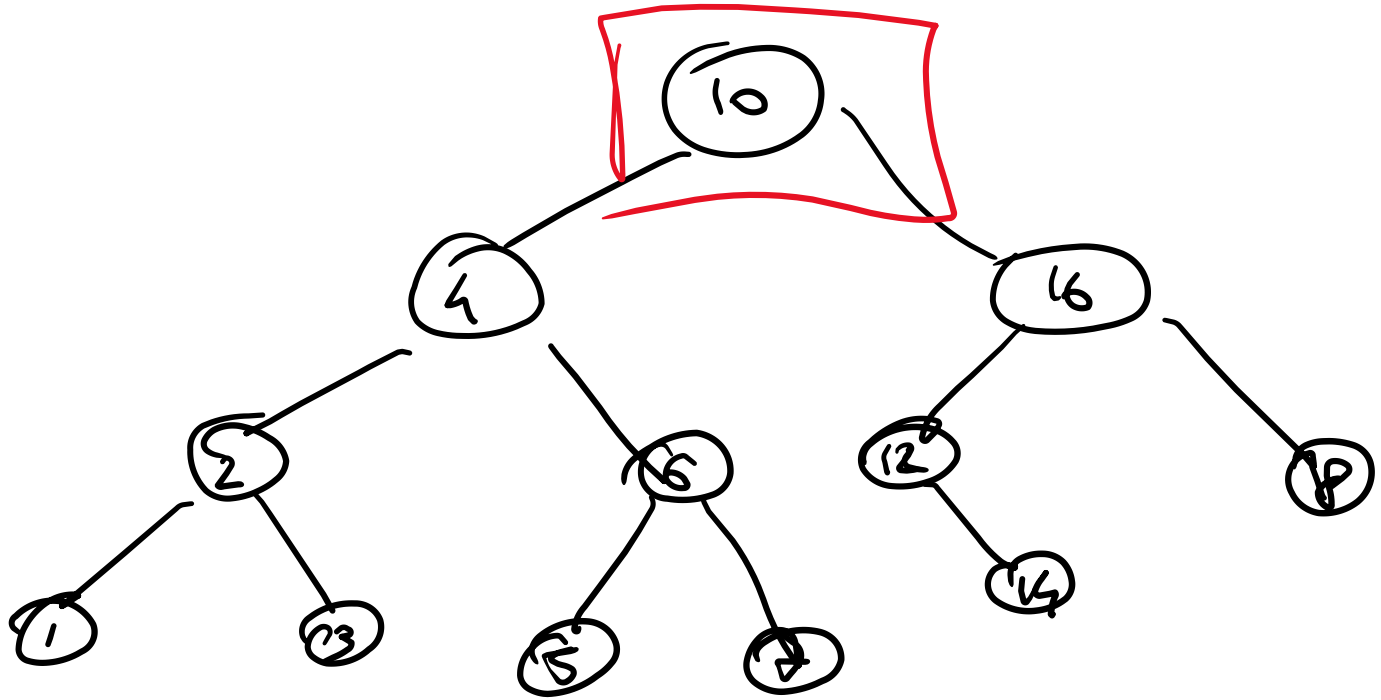
insert (8)

insert (5)



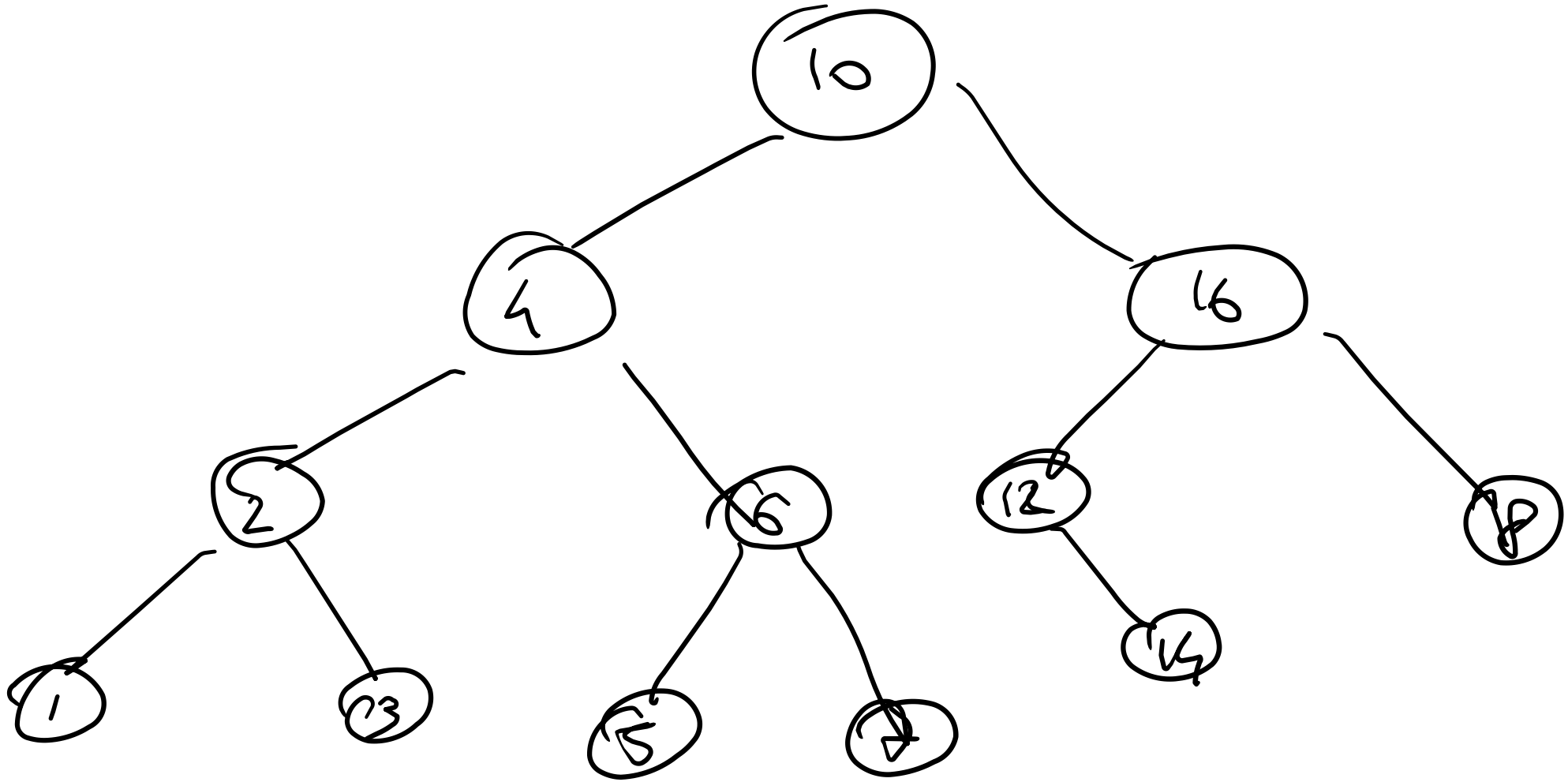
VISITO PRIMA LA RADICE, POI IL SOT. SX
E INFINE IL SOT. DX

VISITA PREORDER



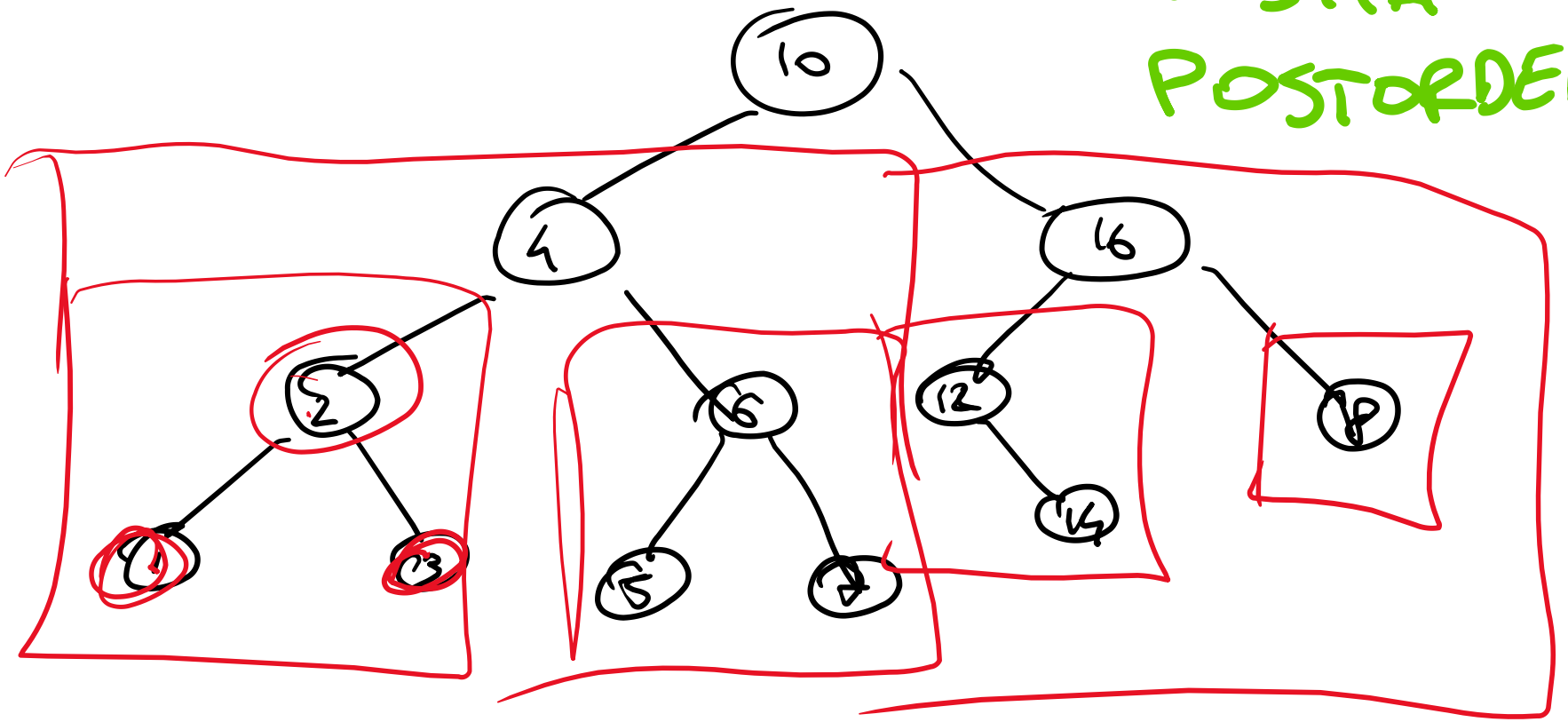
10 - 4 - 2 - 1 - 3 - 6 - 5 - 7 - 16 - 12 - 14 - 18

VISITO PRIMA LA RADICE, POI IL SOT. SX
E INFINE IL SOT. DX



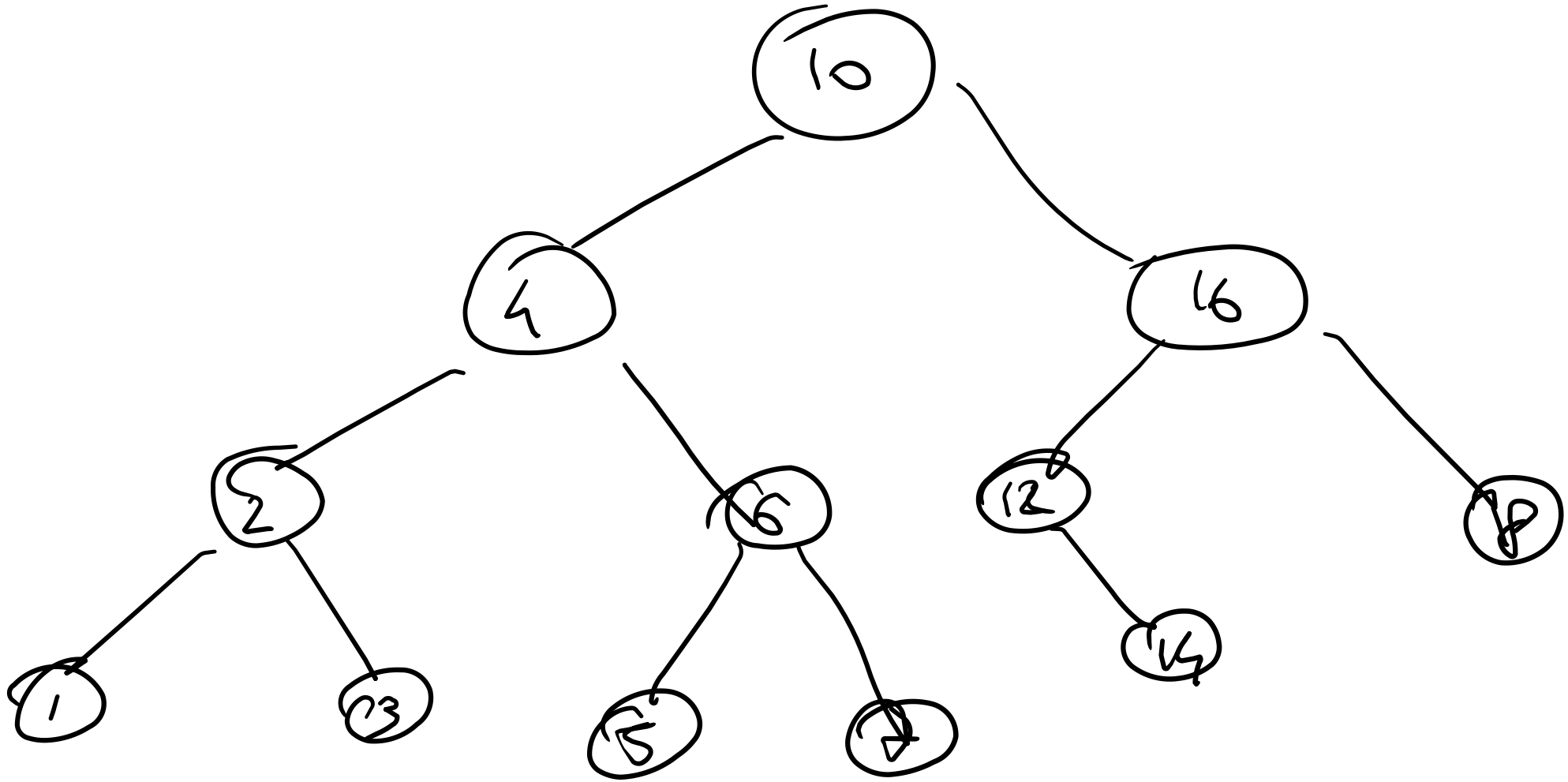
VISITO RICORSIVAMENTE IL SOT. SX, POI, IL SOT. DX E INFINE LA RADICE

VISITA POSTORDER



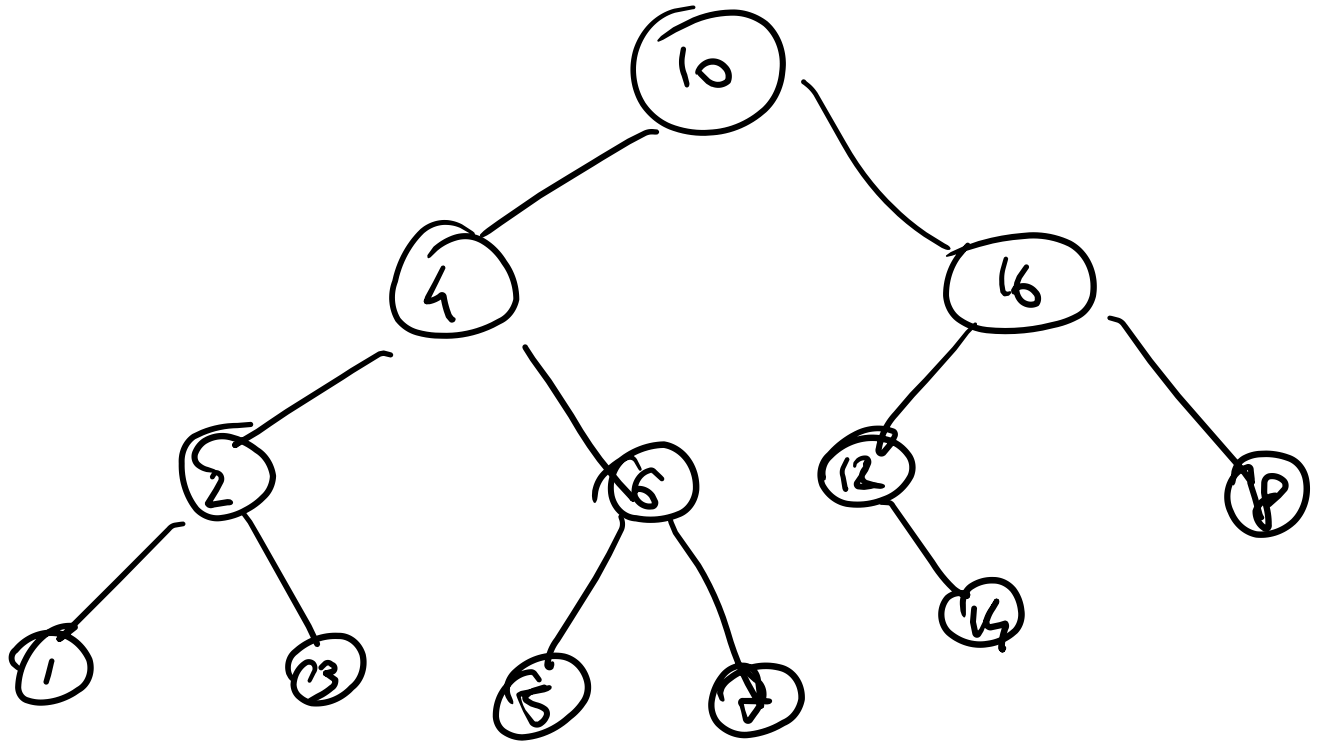
1-3-2-5-8-6-4-14-12-18-16-10

VISITO RICORSIVAMENTE IL SOT. SX, POI, IL SOT. DX E INFINE LA RADICE



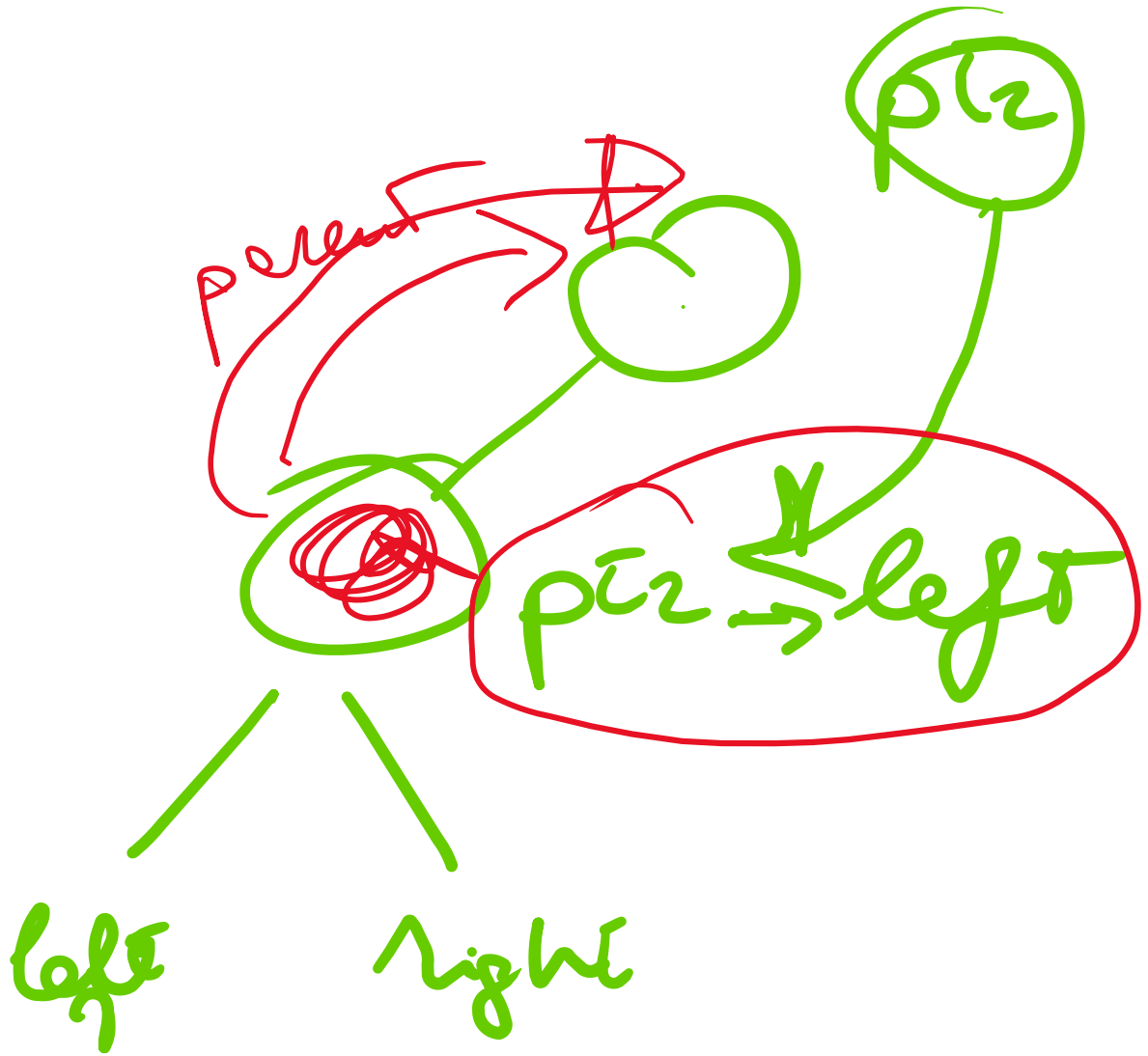
VISITO IL SOTT. SX, POI LA RADICE, E INFINE IL SOTT. DX

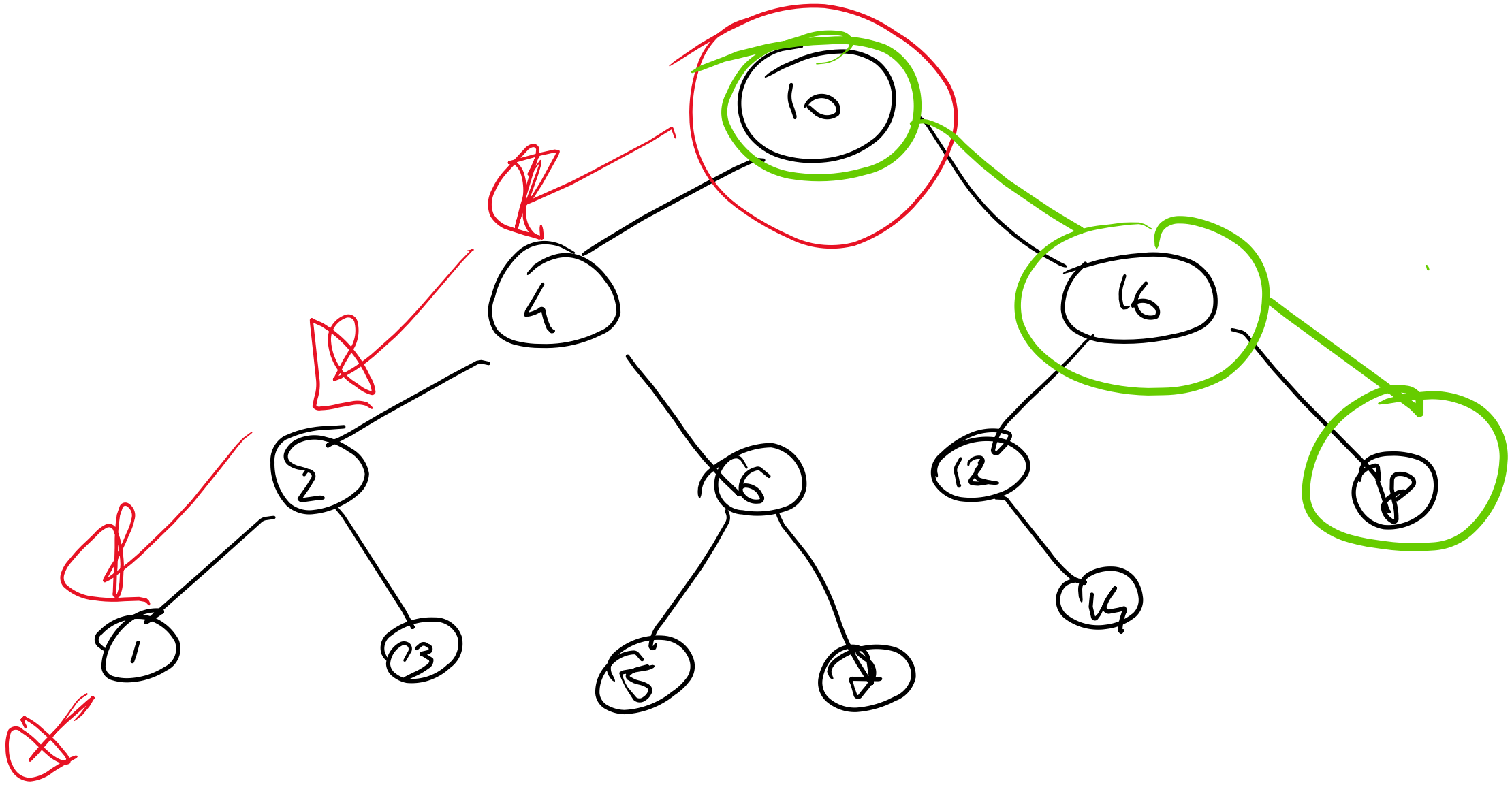
VISITA INORDER



1 - 2 - 3 - 4 - 5 - 6 - 7 - 10 - 12 - 14 - 16 - 18

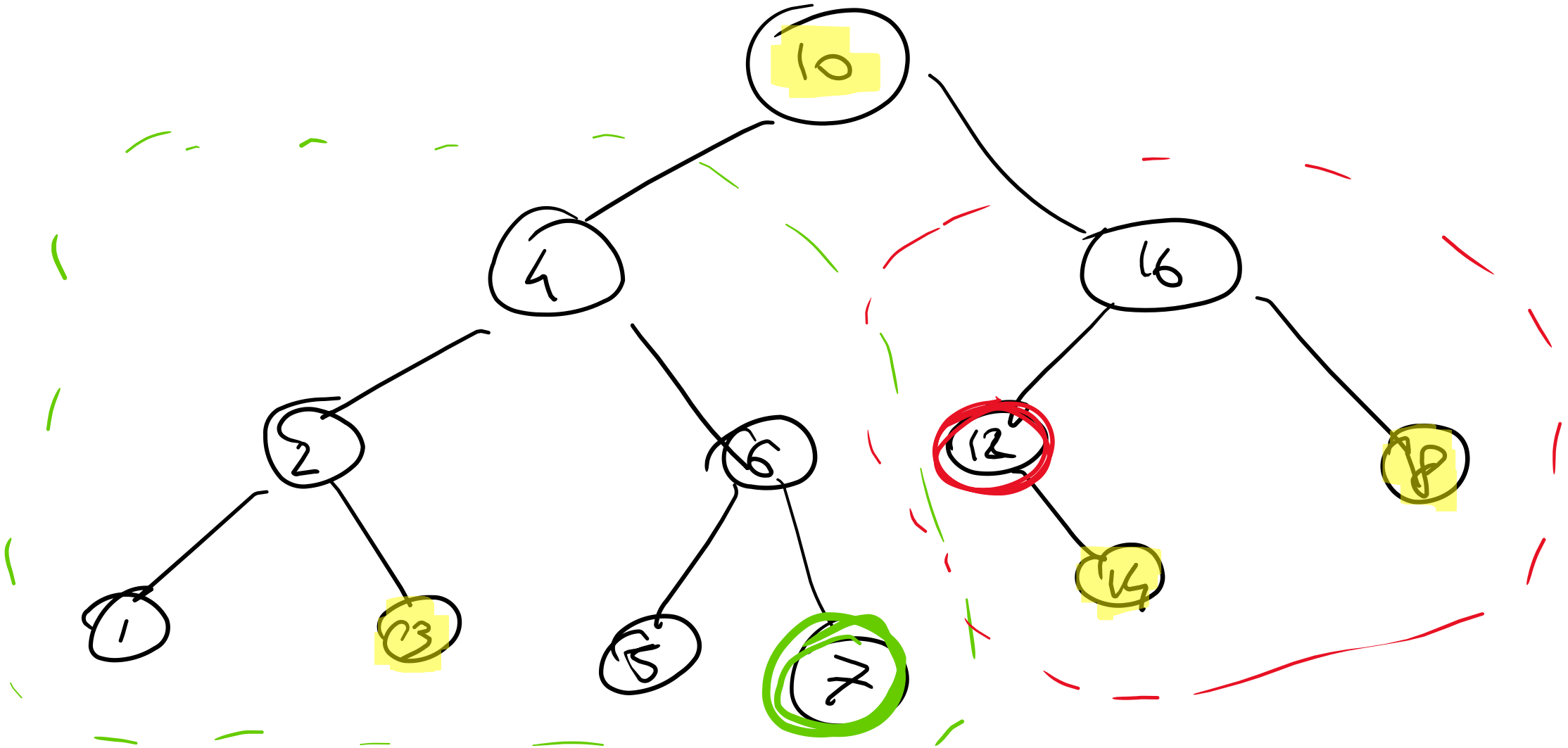
VISITO IL SOT. SX, POI LA RADICE, E INFINE IL SOT. DX


parent
p->r
p->left
left
right

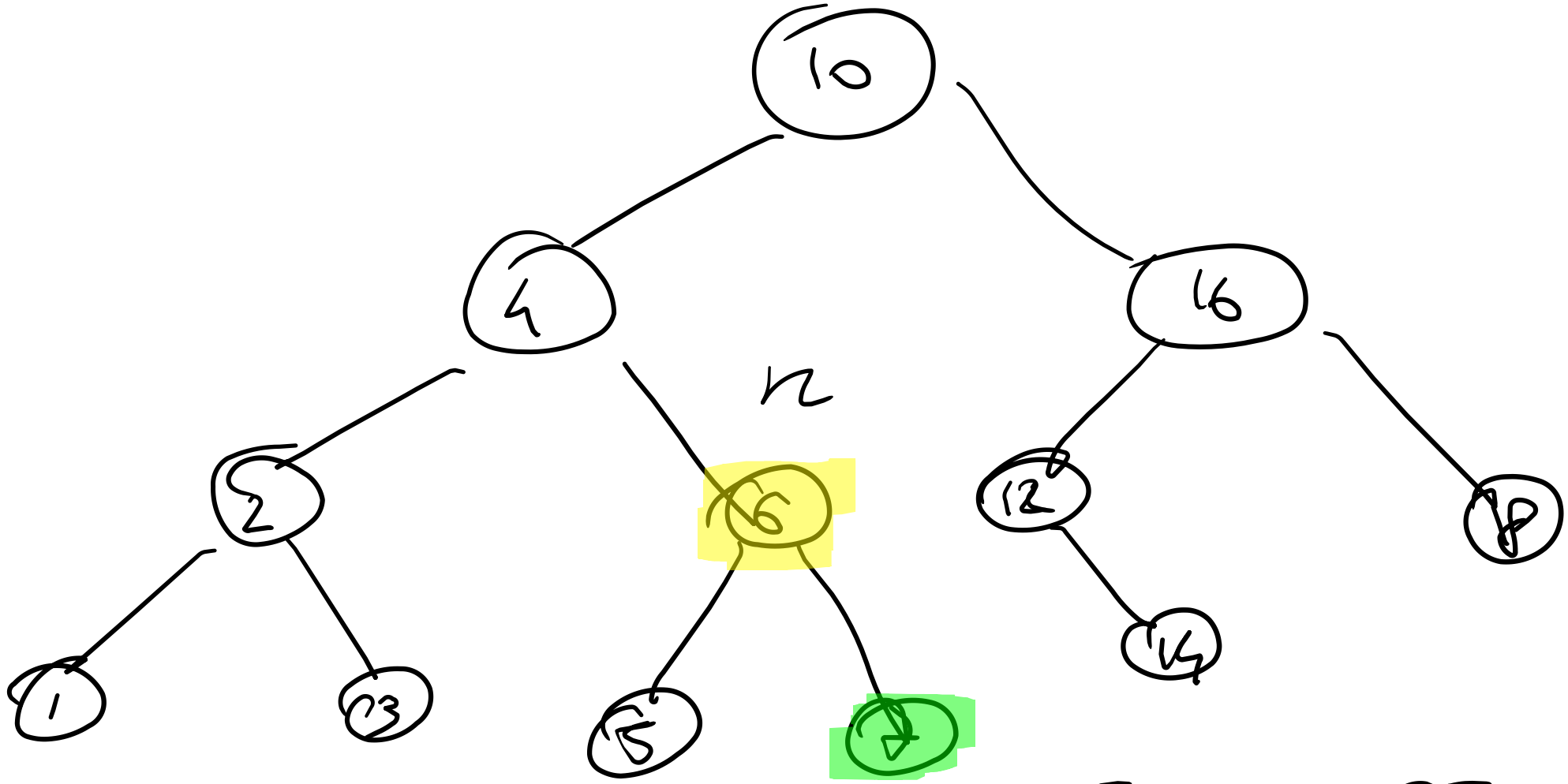


MINIMO

MASSIMO

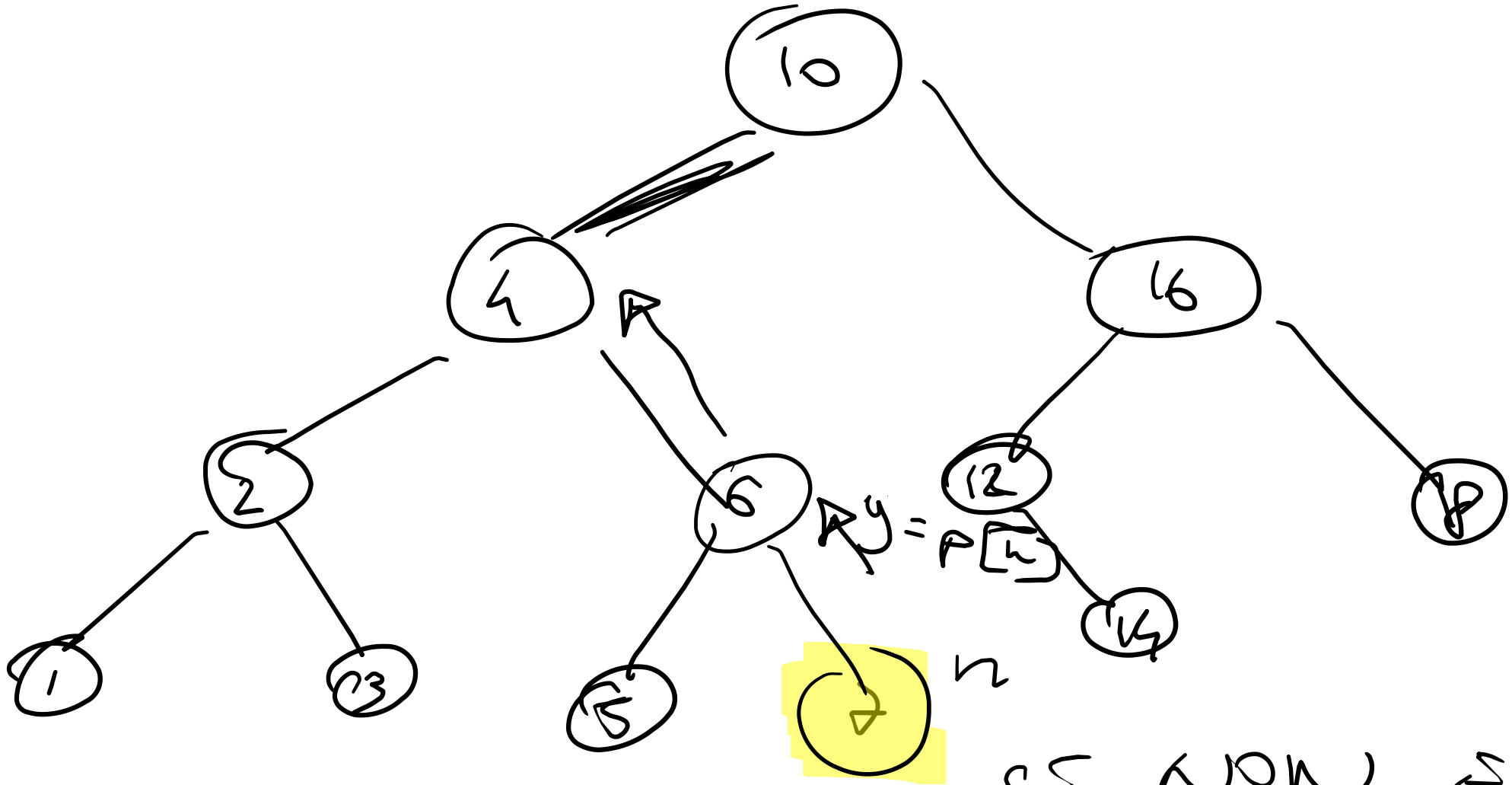


SUCCESSORS — PREDECESSORS

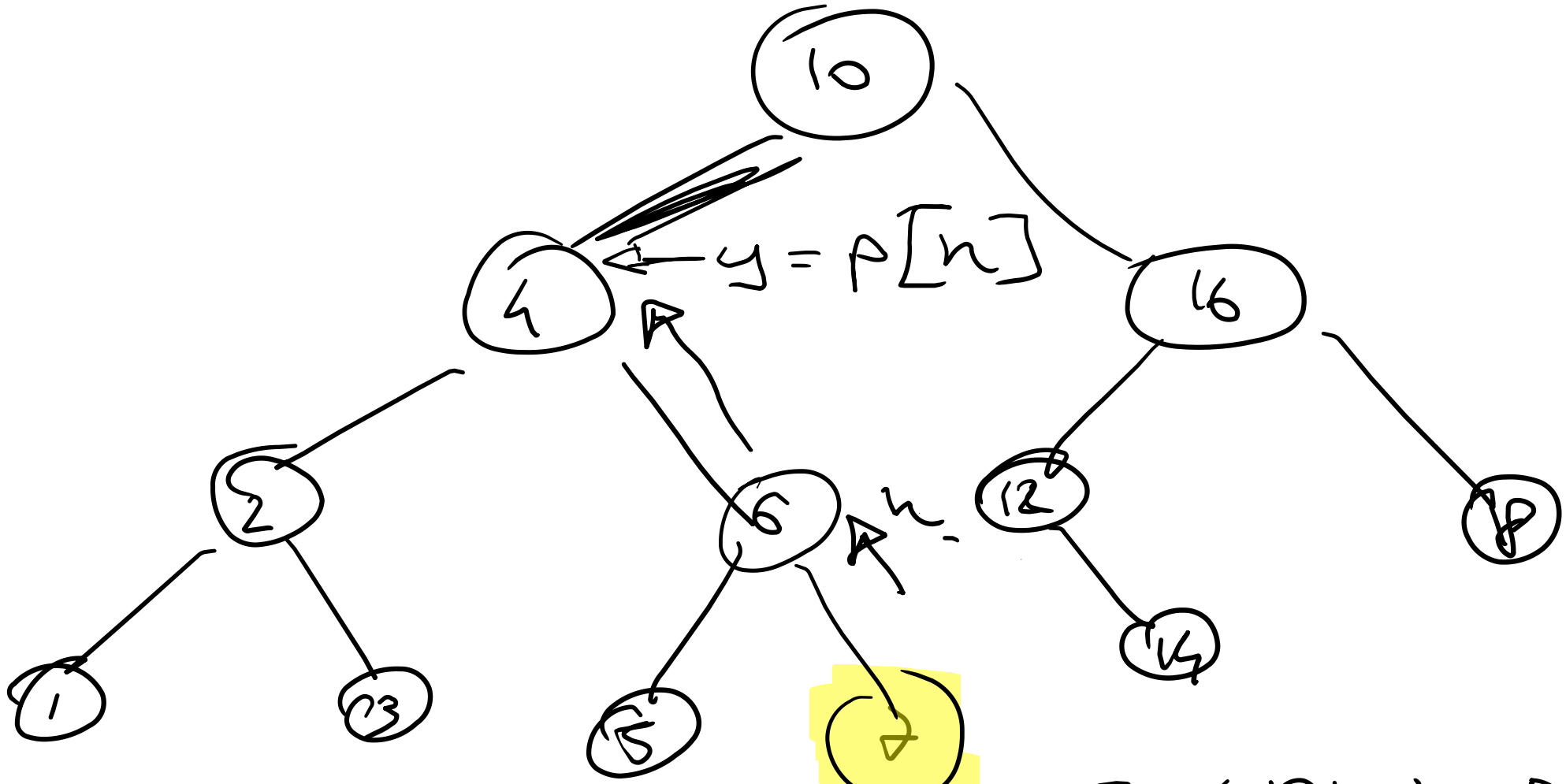


SUCCESSORE (n) →

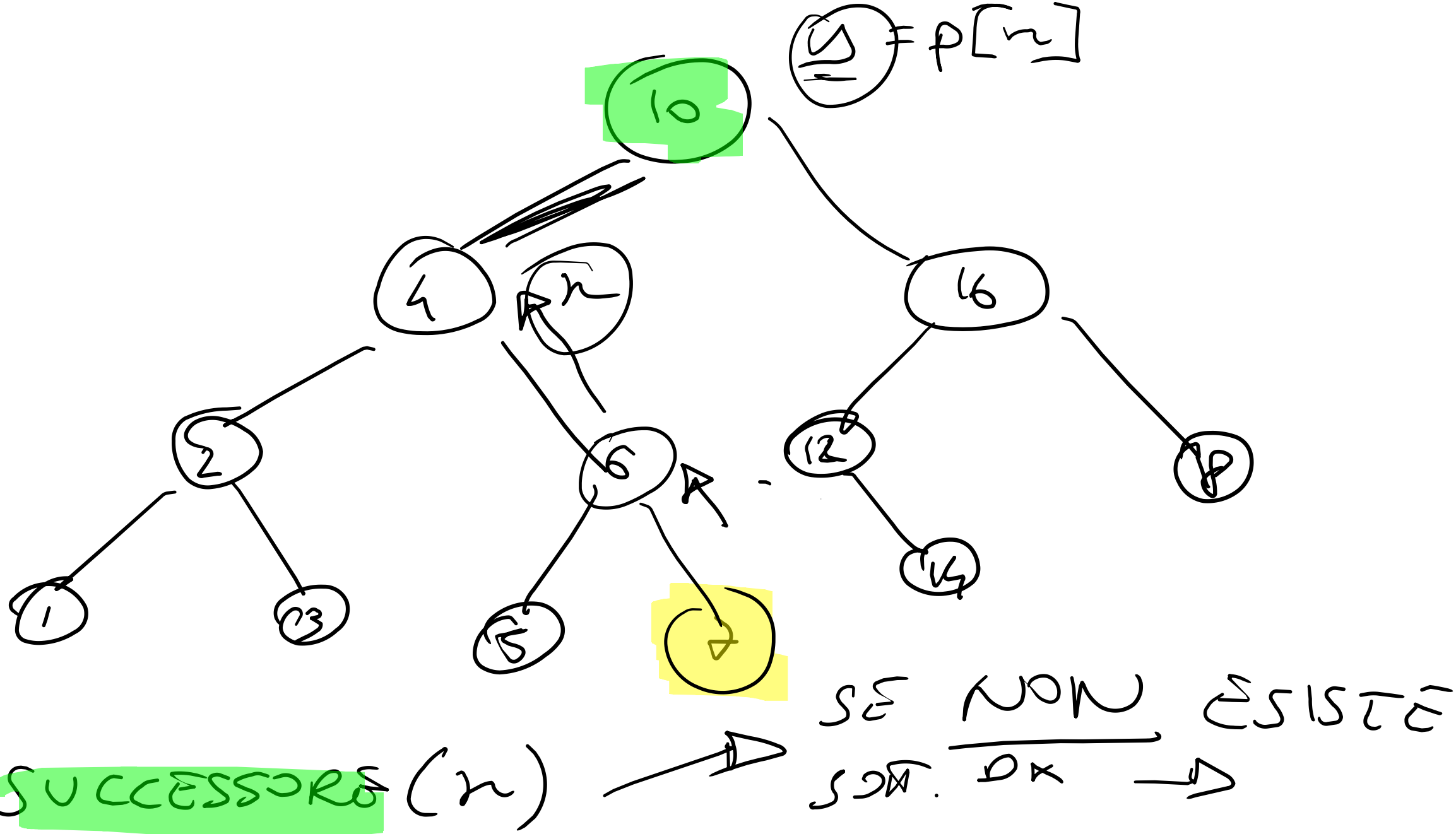
SE ESISTE UN
SOT. DX → IL MIN
DEL SOT. DX



SUCCESSORE (x) → SE NON ESISTE
SOT. DX →



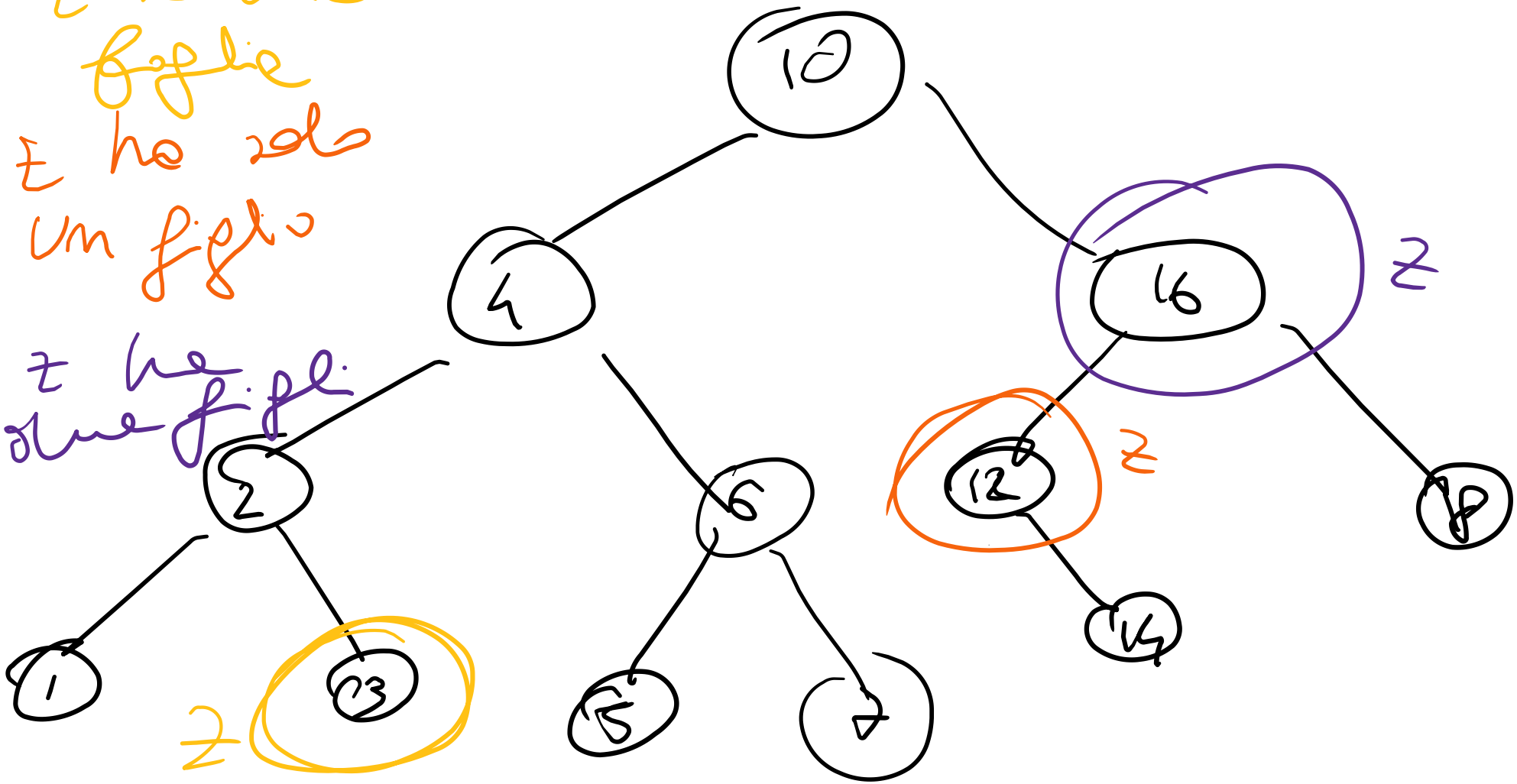
SUCCESSORE (n) → SE NON ESISTE SOT. DX →



SUCCESSORE (x) → SE NON ESISTE DX → SOT.

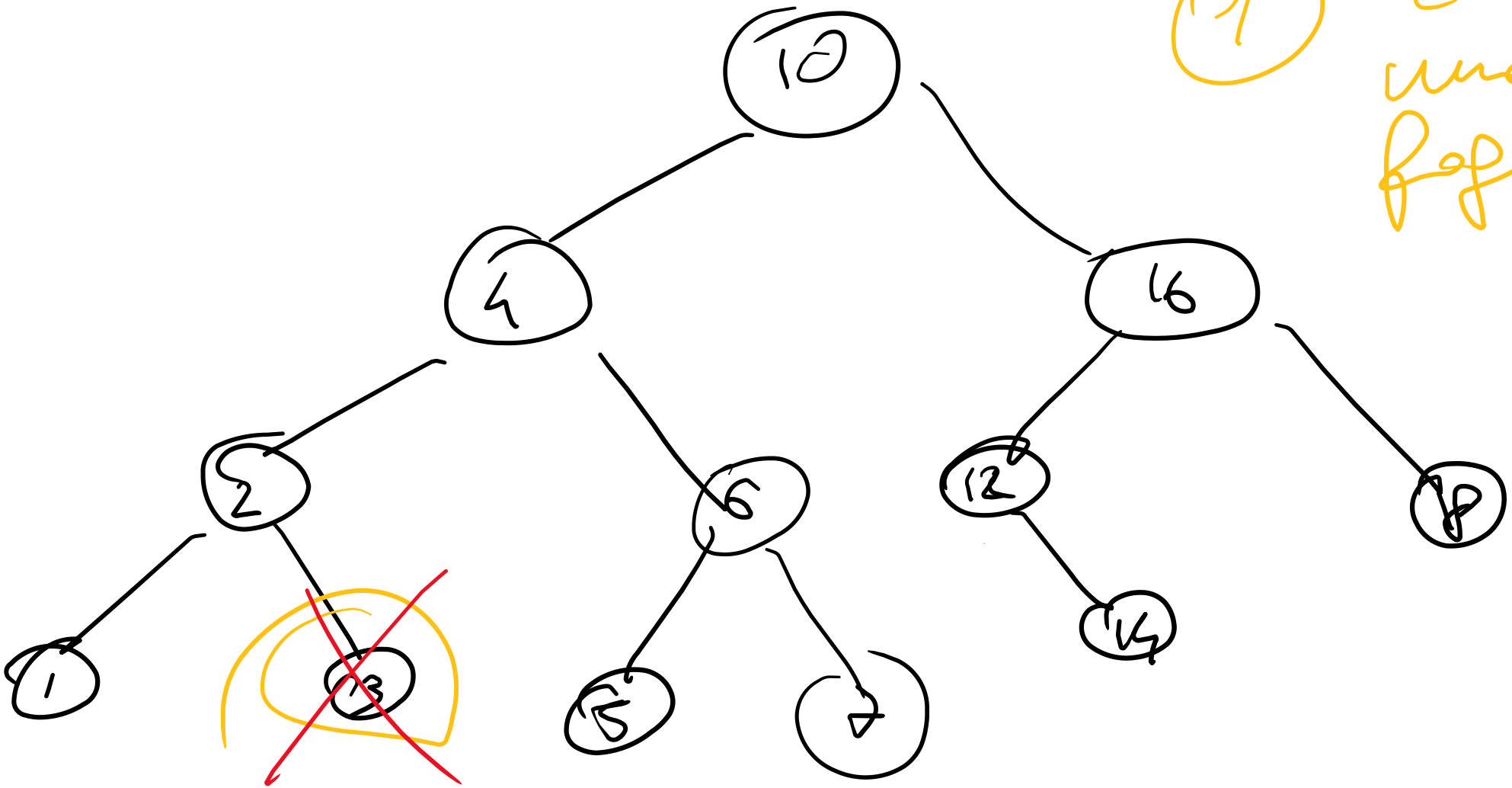
CANCELLAZIONE (T, z)

1. z è una foglia
2. z ha solo un figlio
3. z ha due figli

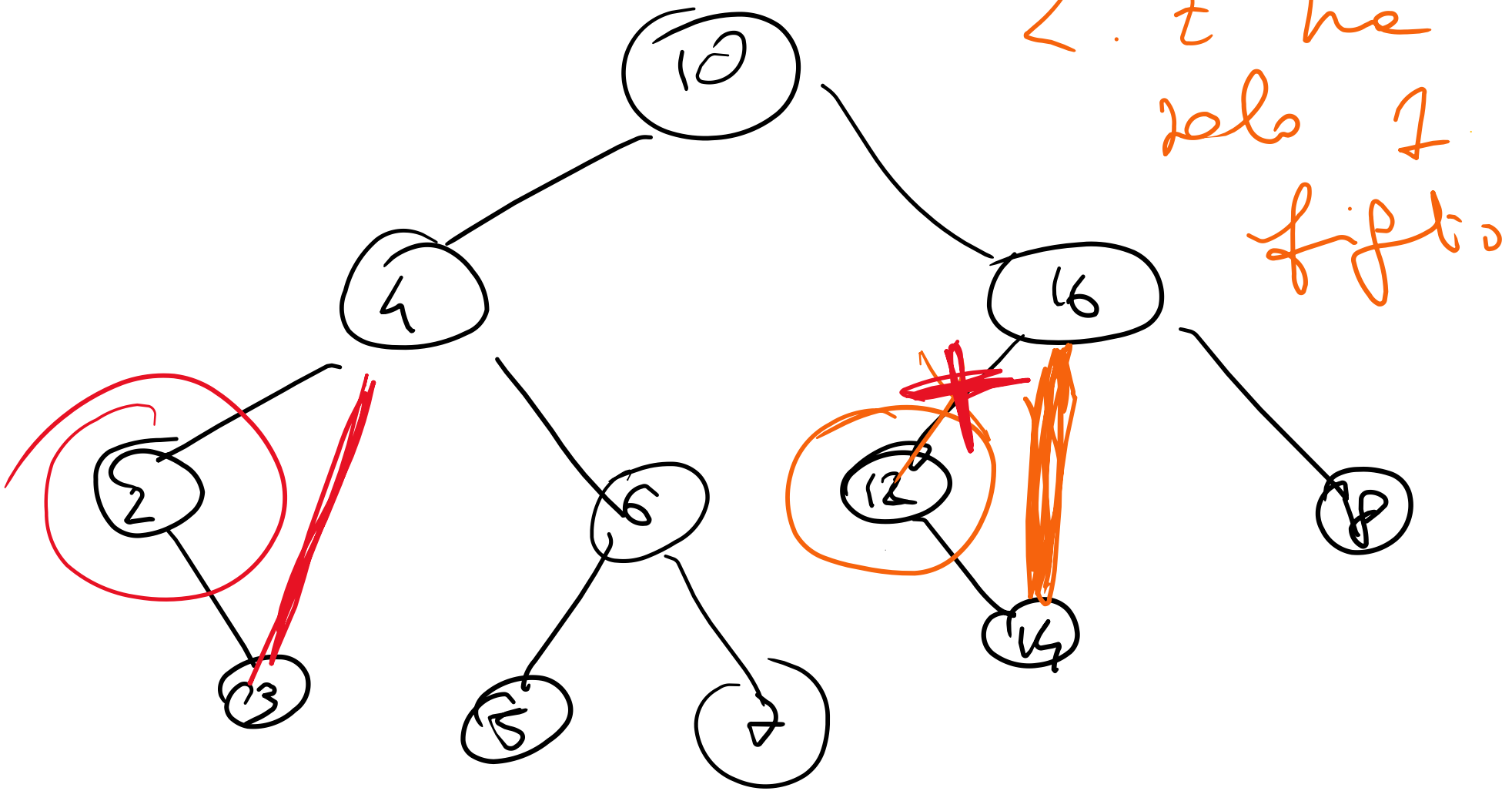


CANCELLAZIONE

1) ti uno foglie



CANCELLAZIONE



2. t ha solo 1 figlio