3.1

Encadenament separat

M = 10 posicions

funció de dispersió $h(x) = x \bmod M$

$h(3441) = 1$

$h(3412) = 2$

⋮

M = 10

0	1	2	3	4	5	6	7	8	9
4830	3441 2001 128181	3412 3722 3202 7812	1983 4893 3313	3874	365		18267	498	1299 999

$\frac{3412}{20} = 170,6$

$170 \times 20 = 3400$

$\frac{3441}{20} = 172,05$

$172 \times 20 = 3440$

M = 20, $h(x) = x \bmod 20$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	3441 2001 128181	3722 3202	1983		365		18267			4830		3412 7812	4893 3313	3874				498	1299 999

$x = 10y + z$

- y par ⇒ $x = 20\frac{y}{2} + z \Rightarrow x \equiv y \pmod{20}$
- y impar ⇒ $x = 20\frac{y-1}{2} + 10 + z \Rightarrow x \equiv y + (10+z) \pmod{20}$

3.2

Encadenament separat

$M = 11$, $h(x) = x \bmod 11$

→ nombre esperat de comparacions en una cerca amb èxit?

→ nombre màxim de comp. en una cerca amb èxit?

Claus:

30, 20, 56, 75, 31, 19 ⇒ $n = 6$

30 mod 11 = 8

20 mod 11 = 9

⋮

11
22
33
44

Taula

$h(30) = 8$

0	1	2	3	4	5	6	7	8	9	10
	56							30	20	
								19	75	
									31	

cerca amb èxit

Nombre esperat de comparacions:

$$1 \cdot \frac{1}{6} + 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 1 \cdot \frac{1}{6} + 2\frac{1}{6} + 3\frac{1}{6} \quad (1,\overline{6})$$

Nombre màxim de comparacions: 3

En general

factor de càrrega

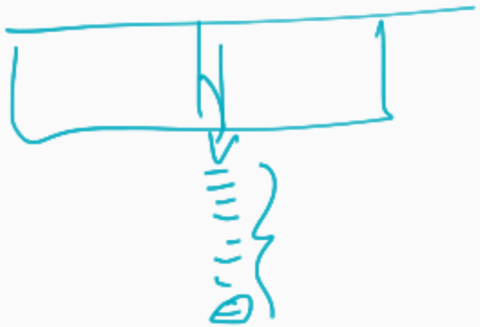
$$\alpha = \frac{n}{M} = 0,54$$

esperat de comparacions

$$\Theta(1+\alpha) = \Theta(1,54)$$

màxim de comparacions

$$\Theta(n)$$



3.3

El renum del BASIC. Un programa en BASIC consisteix en una sèrie d'instruccions numerades en ordre creixent. El control de flux es gestiona a través de les instruccions *GOTO x* i *GOSUB x*, on *x* és un número d'instrucció. Per exemple, el programa següent calcula el factorial d'un número:

```
50 INPUT N
60 LET F = 1
61 LET I = 1
73 IF I=N THEN GOTO 99
76 LET F = F*I
80 LET I = I+1
81 GOTO 73
99 PRINT F
```

El procediment *RENUM* renumera les instruccions del programa de forma que les línies vagin de 10 en 10. Per exemple, després de renumerar, el programa anterior queda:

```
10 INPUT N
20 LET F = 1
30 LET I = 1
40 IF I=N THEN GOTO 80
50 LET F = F*I
60 LET I = I+1
70 GOTO 40
80 PRINT F
```

Com implementar RENUM en temps lineal?

Fem una taula de dispersió amb

CLAU = núm. de línia antic

INFO = núm de línia nou

Mètode?

lineal + lineal = lineal

3.4 **Tots diferents.** Dissenyeu i analitzeu un algorisme que utilitzi taules de dispersió per comprovar que tots els elements d'una llista són diferents.

Tenim una llista l de tipus vector<int>

M
n

```
bool diferents (const vector<int> & l) {
  Dictionary <int,bool> D;
  int i = 0;
  while (i < l.size() and not D.contains(l[i])) {
    D.assign(i,true);
    i++;
  }
  return (i == l.size());
}
```

cas pitjor

$$\theta(M) + \sum_{i=0}^{n-1} \theta(n) =$$

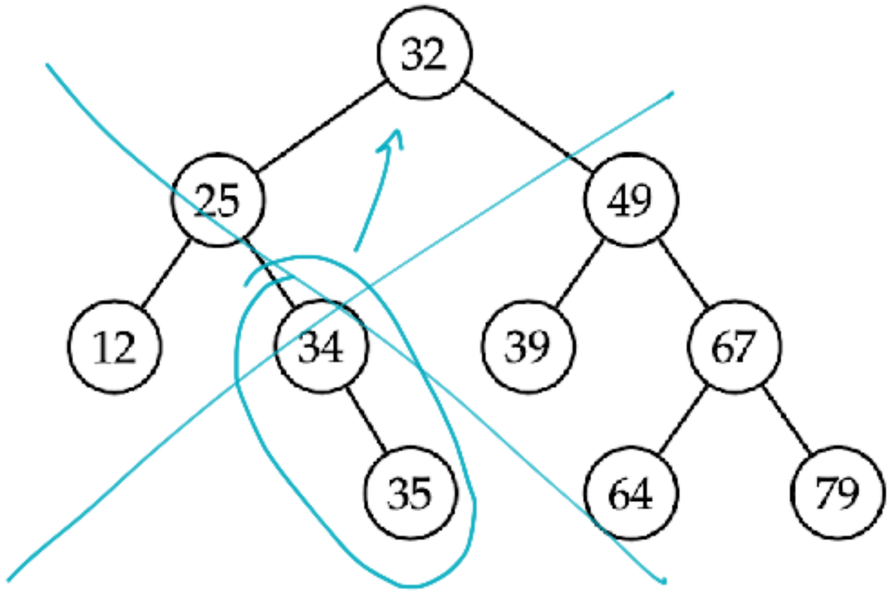
$$\theta(M) + \theta(n^2) = \theta(M+n^2)$$

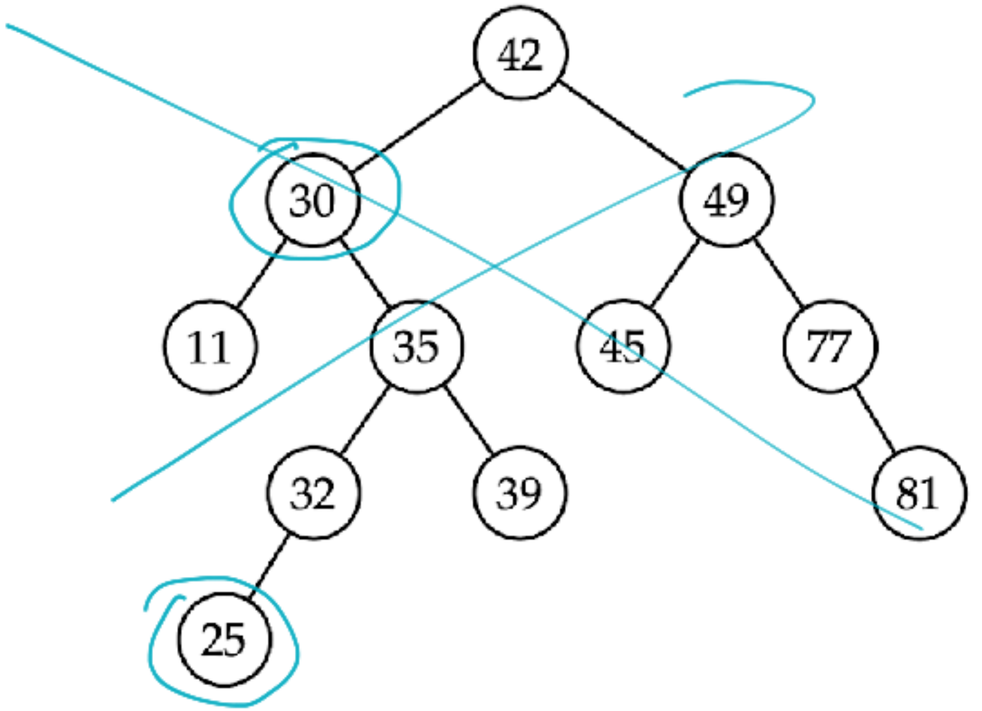
cas millor

$$\sum_{i=0}^{n-1} \theta(1) = \theta(n)$$

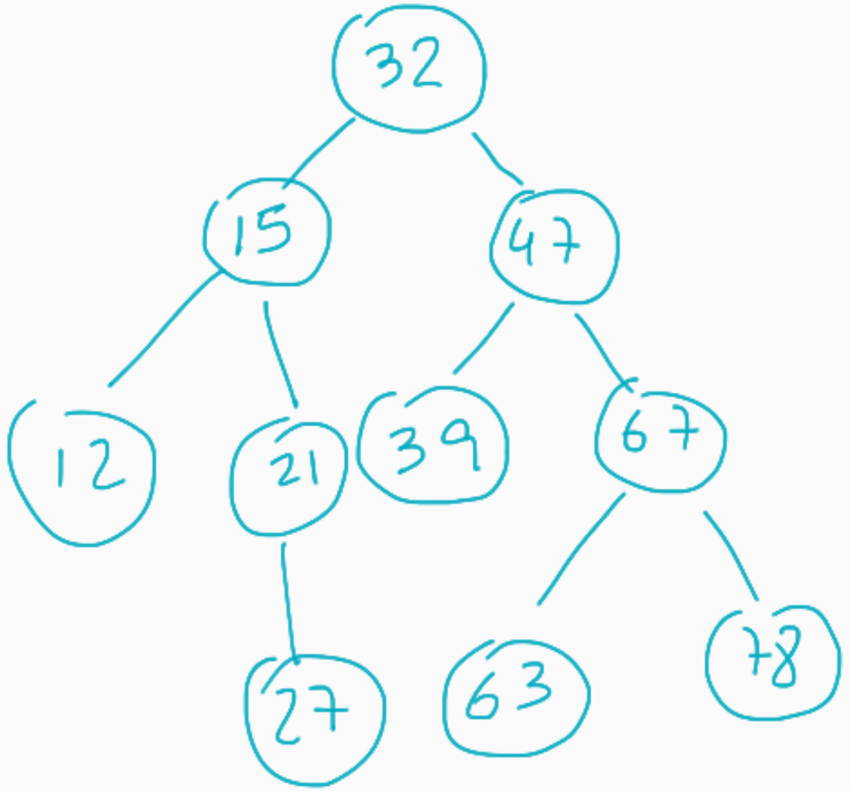
3.5

L'ABC dels ABCs. Digueu si els arbres següents són arbres binaris de cerca o no i per què.

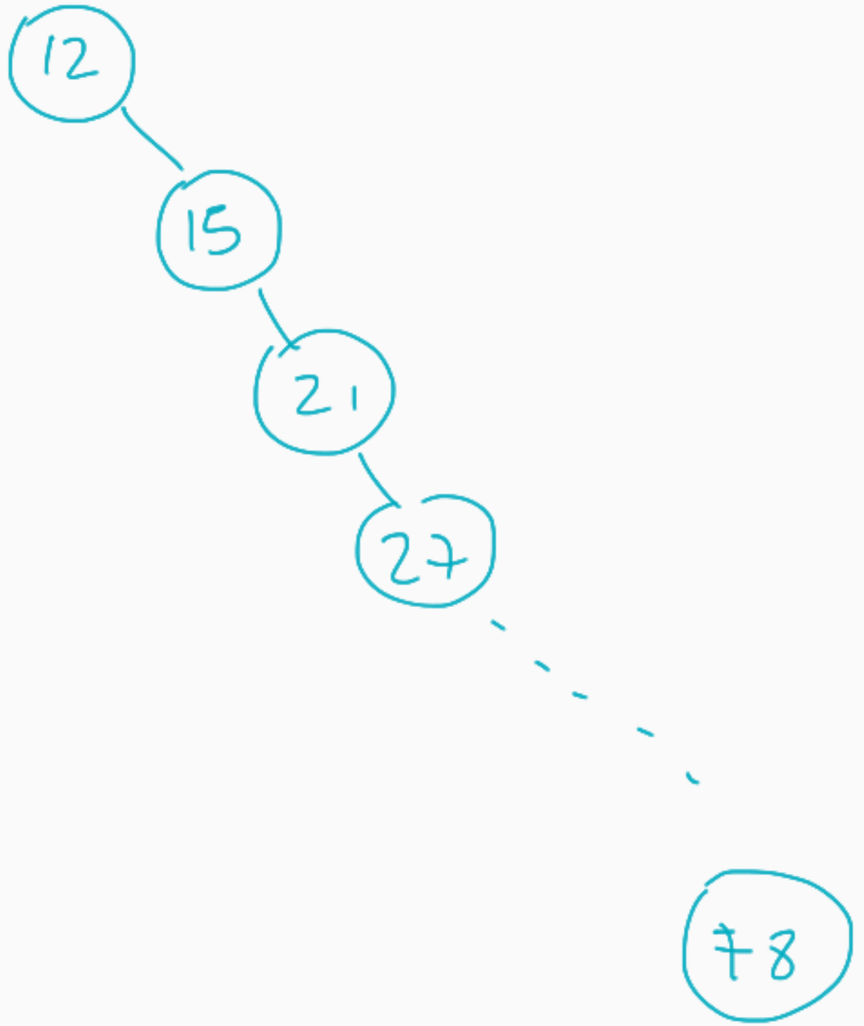




3.6 **Inserir en un ABC.** Partint d'un arbre binari de cerca buit, inseriu amb l'algorisme clàssic, l'una rera l'altra, la seqüència de claus 32, 15, 47, 67, 78, 39, 63, 21, 12, 27.



3.7 **Inserir en un ABC II.** Partint d'un arbre binari de cerca buit, inseriu amb l'algorisme clàssic, l'una rera l'altra, la seqüència de claus 12, 15, 21, 27, 32, 39, 47, 63, 67, 78.

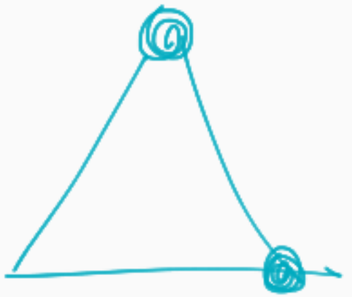


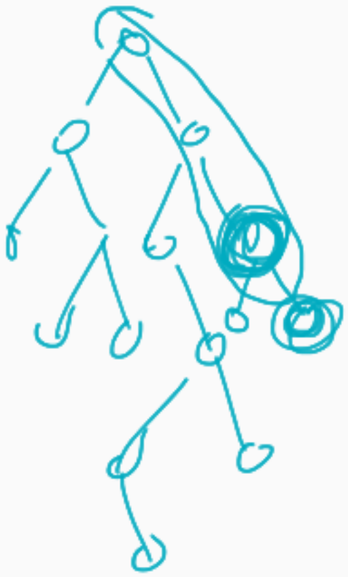
3.8 **Sobre el màxim d'un ABC.** Expliqueu si és cert o no que en un arbre binari de cerca no buit, l'element màxim pot tenir fill esquerre però no pot tenir fill dret.

ABC

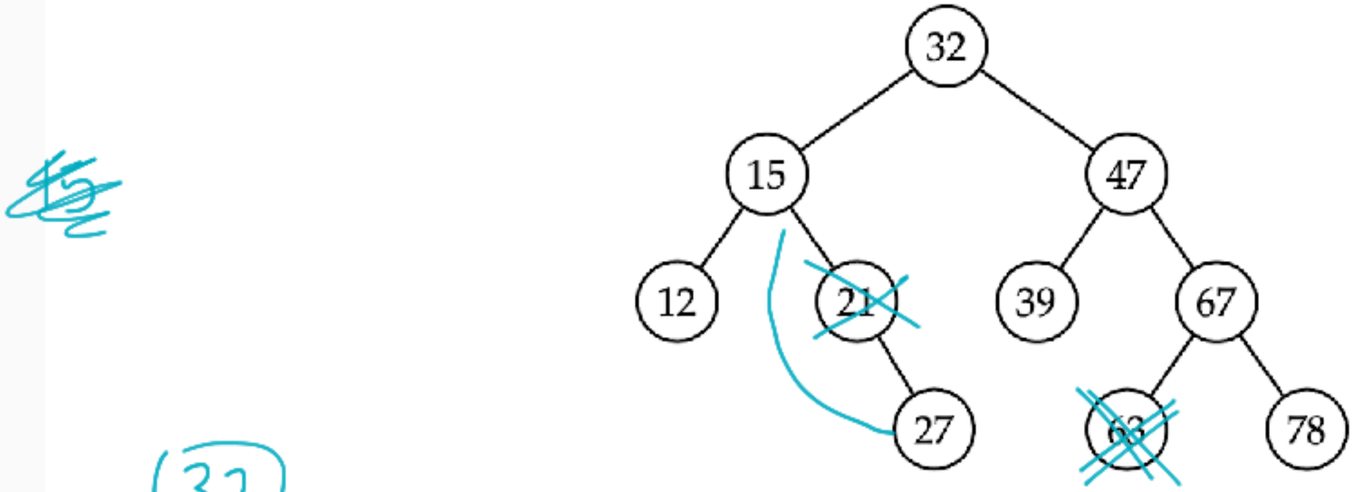
Si x té fill dret, y $\Rightarrow$ $y > x$

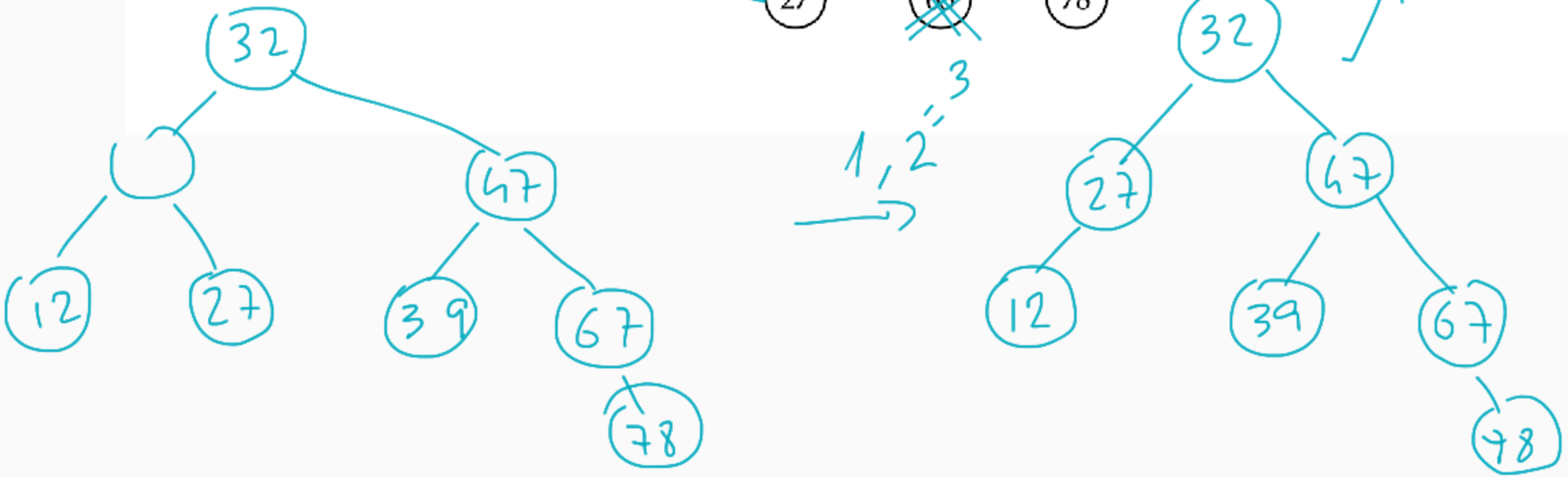
x no és el màxim $\Leftarrow$

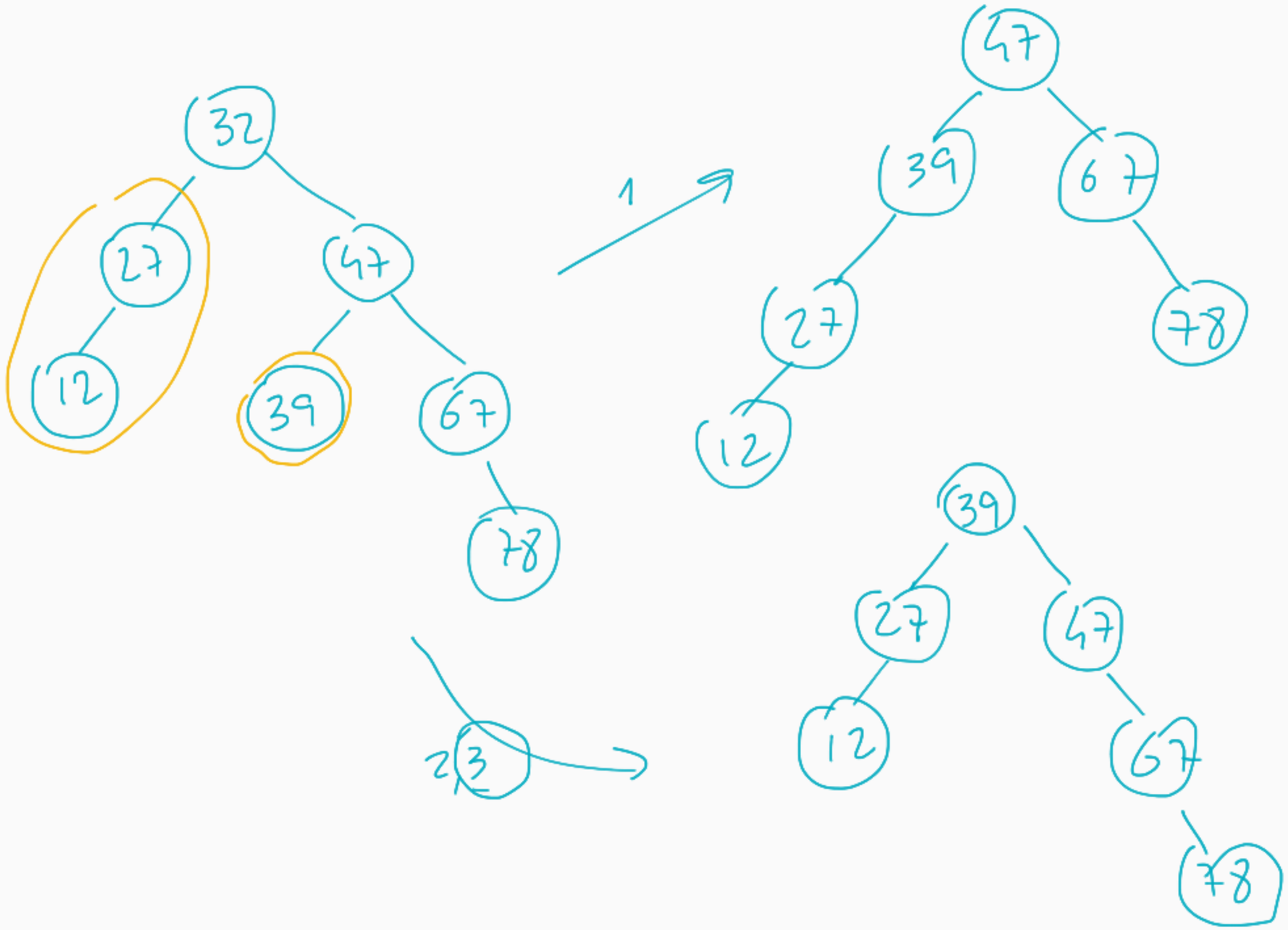




3.9 **Esborrar en un ABC.** Partint de l'arbre binari de cerca següent, elimineu les claus 63, 21, 15, 32, l'una rera l'altra. Expliqueu quin algorisme heu usat per eliminar les claus.






32
27
47
12
39
67
78
1
47
39
67
27
78
12
2
3
39
27
47
12
67
78

3.10 **Inordre d'un ABC.** Demostreu que el recorregut en inordre d'un arbre binari de cerca visita els elements en ordre creixent.

A és un ABC

notació:
- se(A): subarbre esq. d'A
- sd(A): " dret "
- arrel(A): arrel d'A (dada)
- inordre(A): llista corr. al recorregut inordre d'A
- $l_1 \cdot l_2$: concatenació de les llistes l_1 i l_2

1, 3, 4, 5, 7, 9

Def.

$$\text{inordre}(A) = \begin{cases} () & \text{si } |A| = 0 \\ \text{inordre}(se(A)) \cdot \text{arrel}(A) \cdot \text{inordre}(sd(A)) \end{cases}$$

Prop. A és un ABC $\Longrightarrow$ inordre(A) està ordenada

$P(n) \equiv \forall A \ |A|=n :$

Prop. [A és ABC => inordre(A) està ordenada]

Dem.

Inducció en $|A|=n$.

Inducció completa

$P(0), P(1) \ldots P(n-1)$

$n=0$: inordre(A) = ( ) està ordenada

$\forall x, y$ si x apareix abans que y

$\Rightarrow x \leq y$

$n>0$: inordre(A) = inordre(se(A)) . arrel(A) . inordre(sd(A))

$|se(A)|, |sd(A)| < n$

h.i. h.i.

inordre(se(A)) està ord.

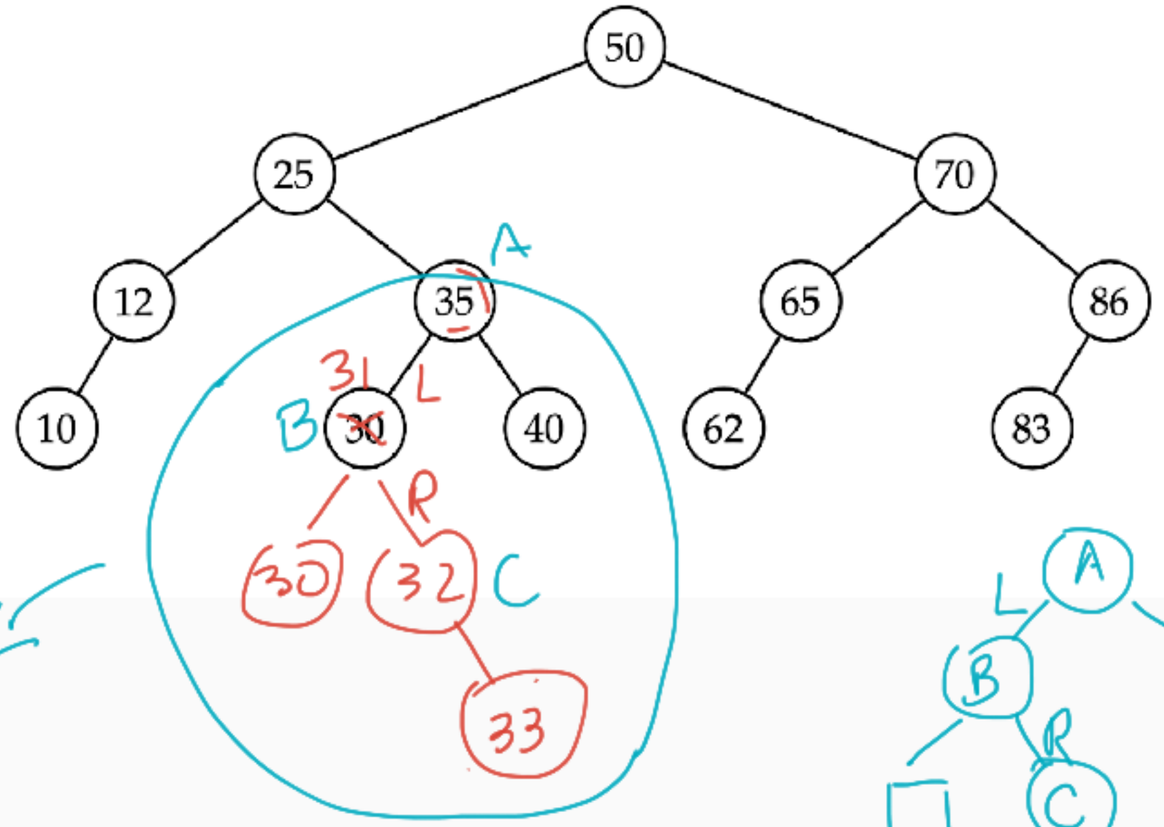
arrel(A)

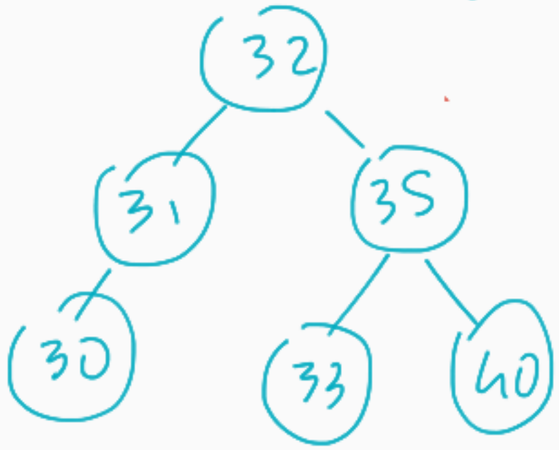
inordre(sd(A)) .. ..

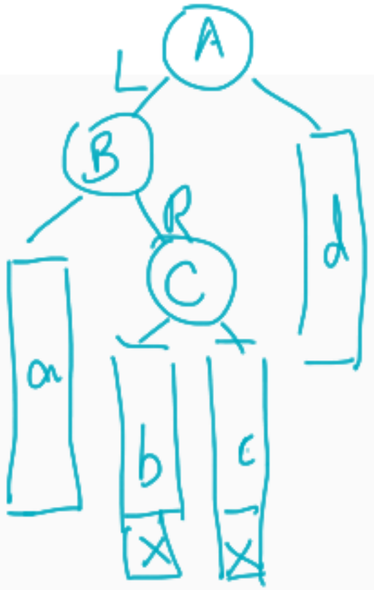
ord

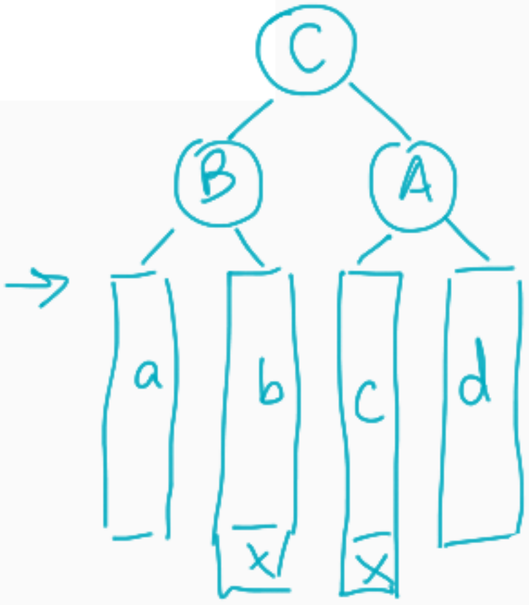
n

3.23 **Inserir en un AVL.** Doneu els tres arbres AVL resultants d'afegir les claus 31, 32 i 33 l'una després de l'altra en l'arbre AVL següent:









3.24 **Inserir i esborrar en un AVL.** Doneu els quatre arbres AVL resultants d'afegir les claus 18 i 12 i d'esborrar les claus 7 i 30 a l'arbre AVL següent (apliqueu cada operació a l'arbre obtingut anteriorment):