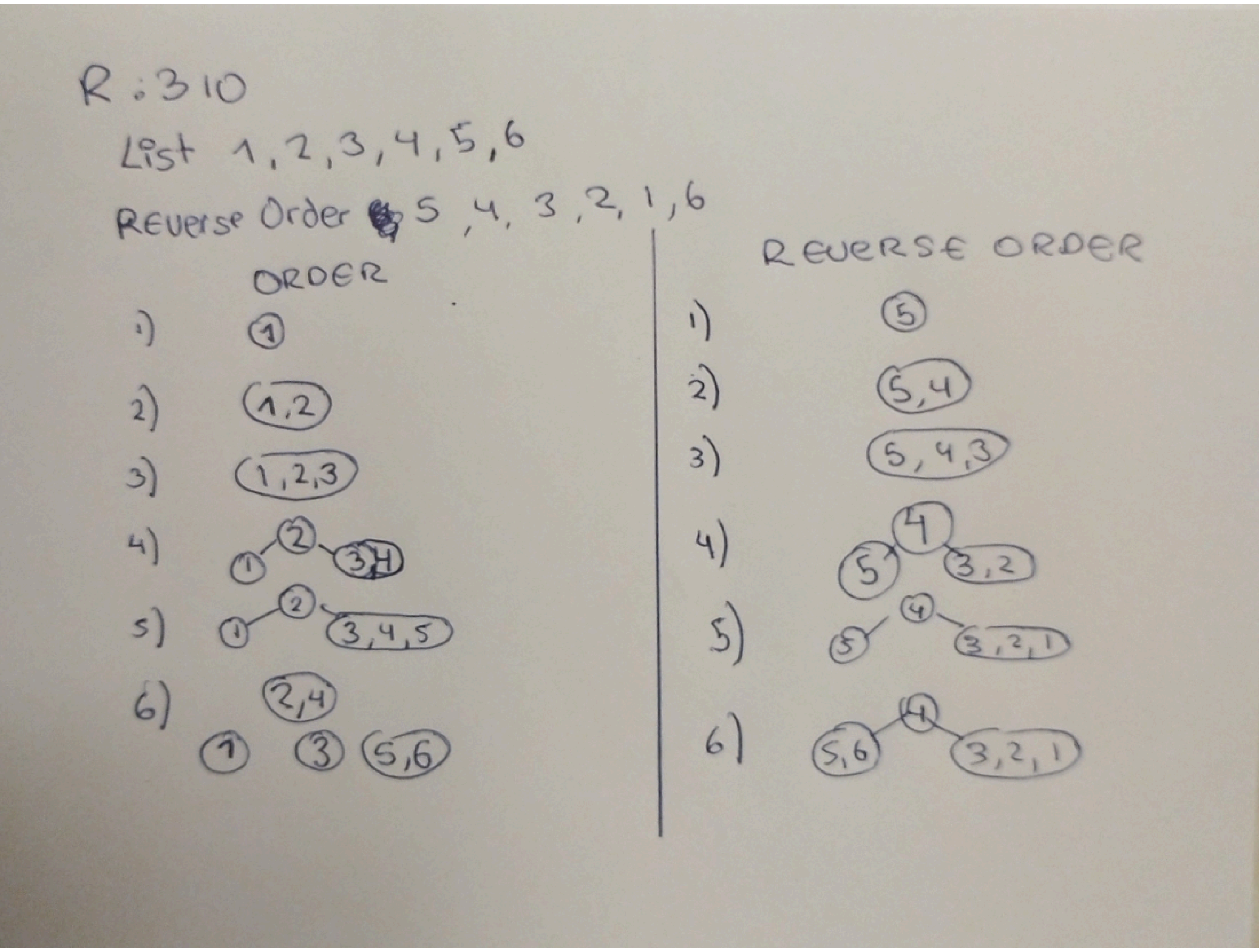


R : 310

List 1, 2, 3, 4, 5, 6

Reverse Order 5, 4, 3, 2, 1, 6

ORDER

1) ①

2) (1, 2)

3) (1, 2, 3)

4)
$$\begin{tikzcd}
 & 2 \arrow[dl, dash] \arrow[dr, dash] & \\
1 & & 3,4
\end{tikzcd}$$

5)
$$\begin{tikzcd}
 & 2 \arrow[dl, dash] \arrow[dr, dash] & \\
1 & & 3,4,5
\end{tikzcd}$$

6)
$$\begin{tikzcd}
 & 2,4 & \\
1 & 3 & 5,6
\end{tikzcd}$$

REVERSE ORDER

1) ⑤

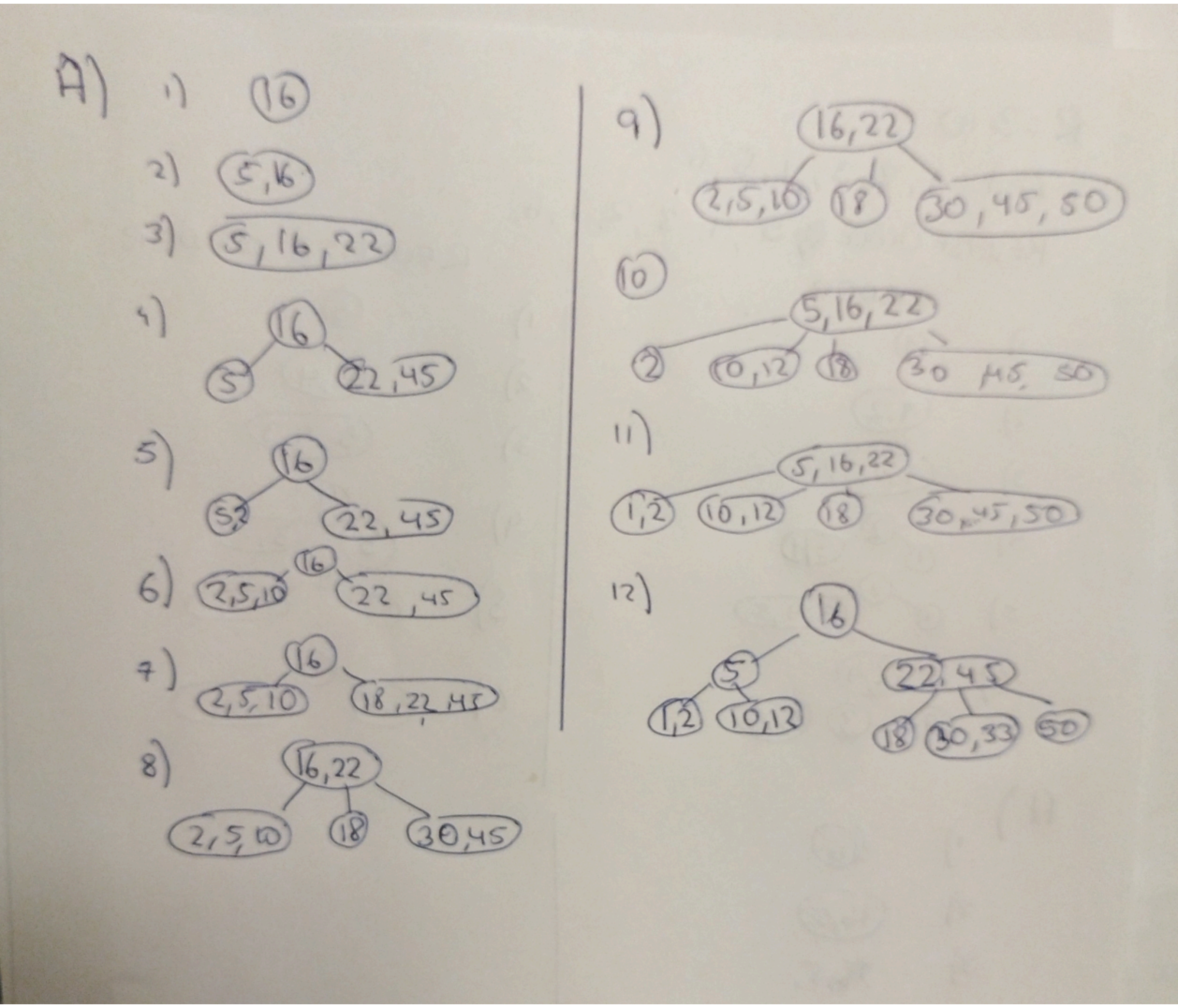
2) (5, 4)

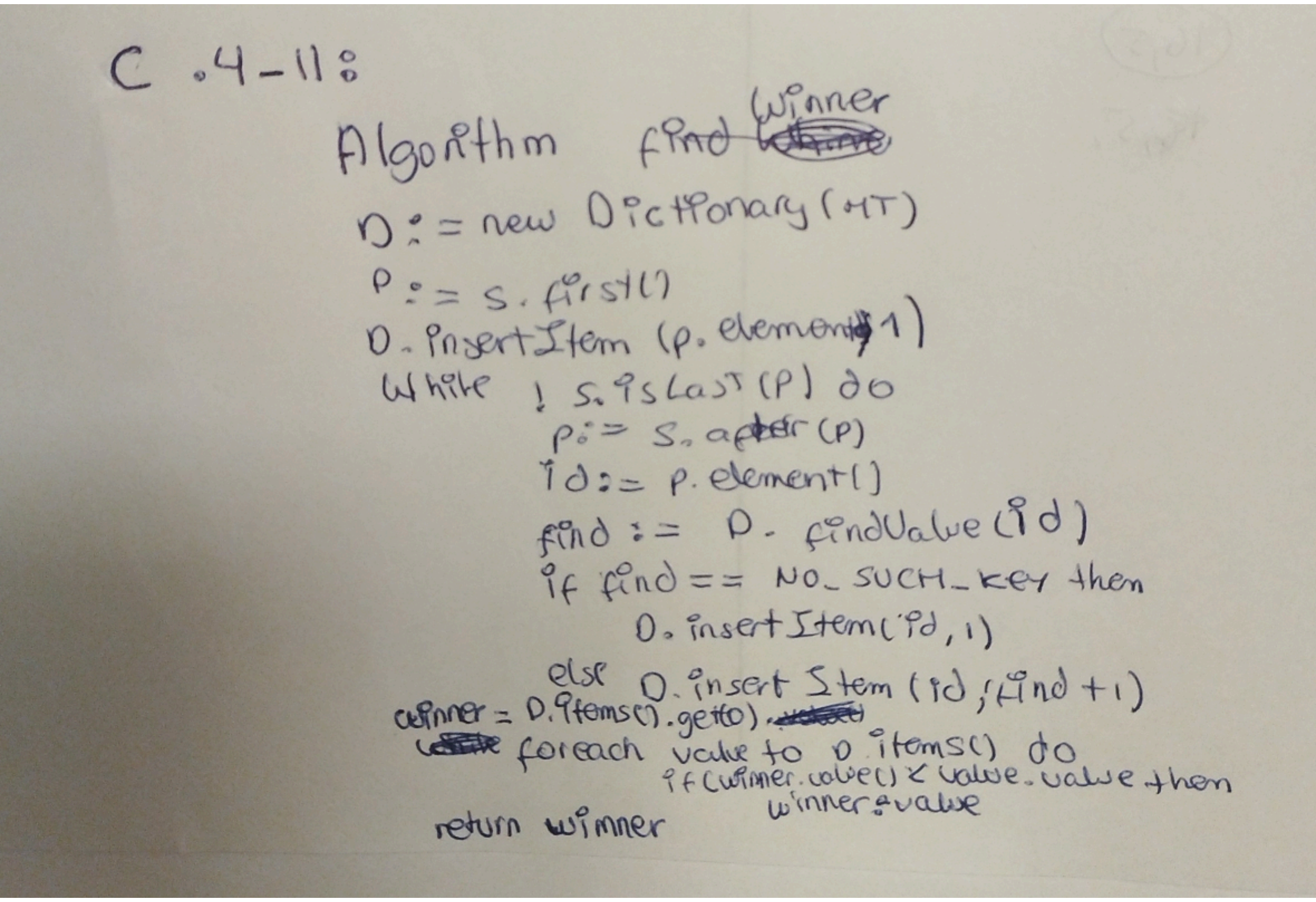
3) (5, 4, 3)

4)
$$\begin{tikzcd}
 & 4 \arrow[dl, dash] \arrow[dr, dash] & \\
5 & & 3,2
\end{tikzcd}$$

5)
$$\begin{tikzcd}
 & 4 \arrow[dl, dash] \arrow[dr, dash] & \\
5 & & 3,2,1
\end{tikzcd}$$

6)
$$\begin{tikzcd}
 & 4 \arrow[dl, dash] \arrow[dr, dash] & \\
5,6 & & 3,2,1
\end{tikzcd}$$

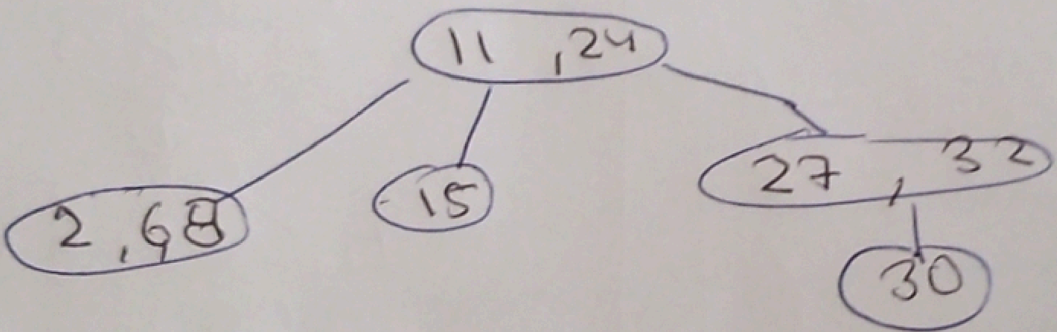

A)
1) 16
2) 5,16
3) 5,16,22
4) 16 / 5 / 22,45
5) 16 / 5,2 / 22,45
6) 16 / 2,5,10 / 22,45
7) 16 / 2,5,10 / 18,22,45
8) 16,22 / 2,5,10 / 18 / 30,45
9) 16,22 / 2,5,10 / 18 / 30,45,50
10) 5,16,22 / 2 / 10,12 / 18 / 30 45, 50
11) 5,16,22 / 1,2 / 10,12 / 18 / 30,45,50
12) 16 / 5 / 22,45 / 1,2 / 10,12 / 18 / 30,33 / 50



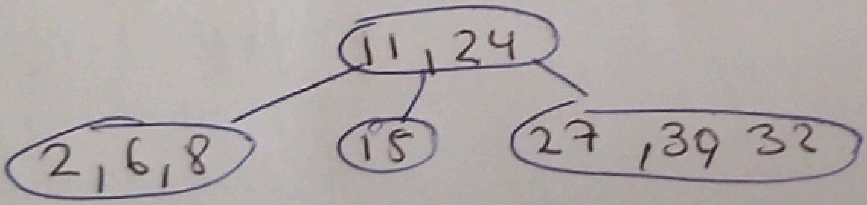
C .4-11:

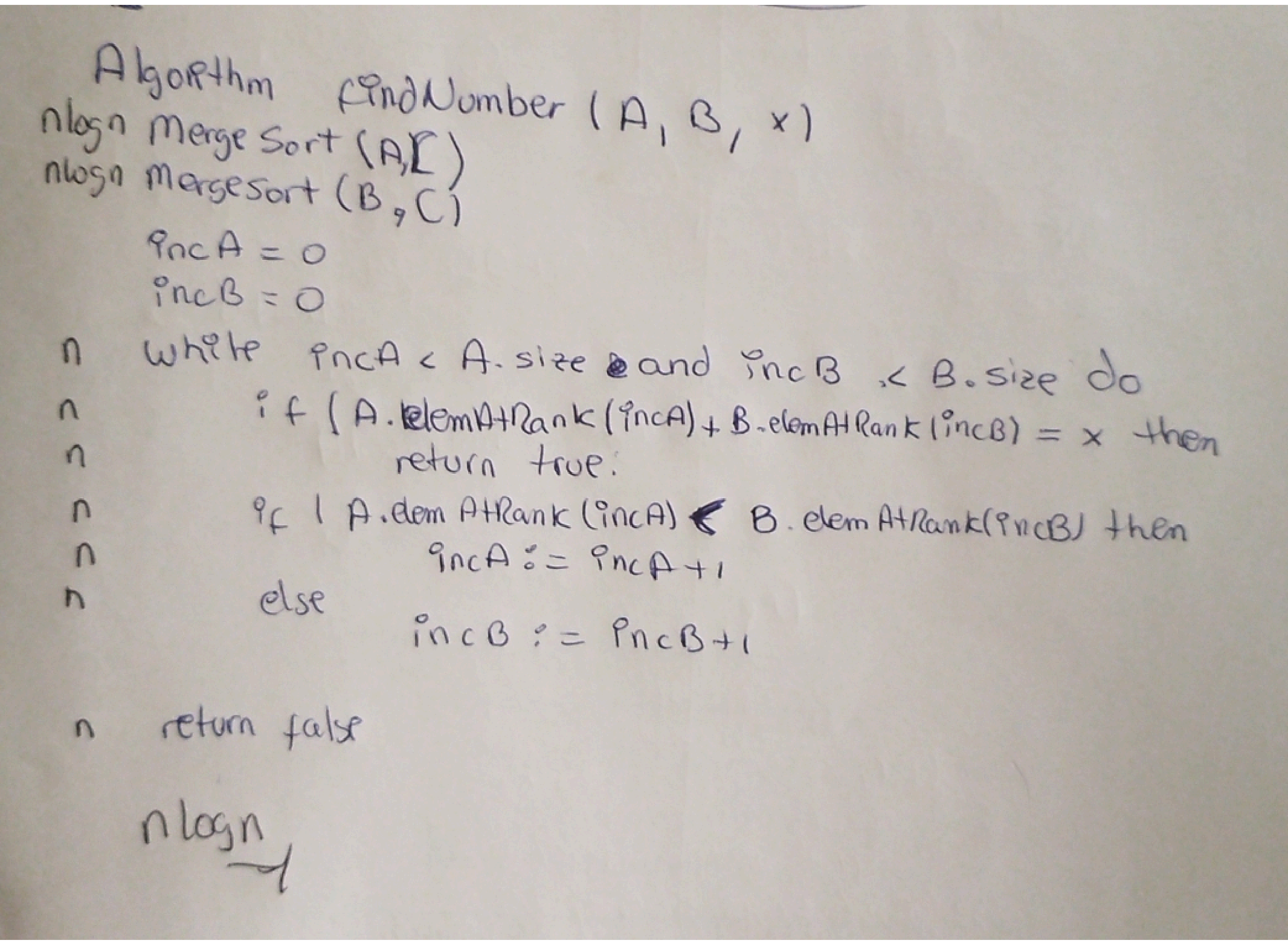
```
Algorithm find ~~[illegible]~~ Winner
D := new Dictionary(HT)
P := S.first()
D.insertItem(p.element(),1)
While ! S.isLast(P) do
    p := S.after(P)
    id := p.element()
    find := D.findValue(id)
    if find == NO_SUCH_KEY then
        D.insertItem(id,1)
    else D.insertItem(id,find+1)
winner = D.items().get(0). [illegible]
foreach value to D.items() do
    if(winner.value() < value.value then
        winner:=value
return winner
```

Why is'not valid ?



Answer: Because each node can contains 3 nodes





```
Algorthm findNumber (A, B, x)
nlogn   MergeSort (A, C)
nlogn   Mergesort (B, C)
          incA = 0
          incB = 0
n       while incA < A.size and incB < B.size do
n            if (A.elemAtRank(incA) + B.elemAtRank(incB) = x then
n                  return true.
n            if (A.elemAtRank(incA) < B.elemAtRank(incB) then
n                  incA := incA + 1
n            else
                  incB := incB + 1

n       return false

nlogn
```