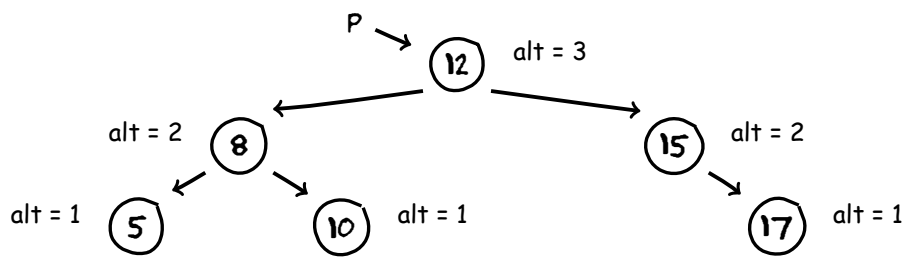
Pré-calculando a altura

Mas, não é preciso percorrer a árvore toda vez que nós quisermos consultar a sua altura.

Quer dizer, a ideia é que cada elemento pode armazenar a altura da (sub)árvore que está abaixo dele — (*no campo* `q->alt`)



Daí, para descobrir a altura da ávore, basta consultar o campo `.alt` da raiz — (*em tempo* $O(1)$)

Mas logo no início, é preciso calcular o valor do campo `.alt` de cada elemento.

Isso pode ser feito com uma pequena modificação do algoritmo do exemplo anterior.

E a boa notícia é que nós não precisamos mais das variáveis `Alt` e `maxAlt`.

Quer dizer, no momento de subir para o pai, cada elemento calcula a sua altura com base nas alturas dos seus filhos.

Para facilitar as coisas, a gente escreve uma função auxiliar para calcular a altura de um elemento

,

```
func calcAlt (q)
     alt = 0
     Se (q->esq != Nulo)    alt = (q->esq)->alt
     Se (q->dir != Nulo)    alt = Max {alt, (q->dir)->alt}
     q->alt  = alt
```

E daí, a coisa fica assim

```
func altura-Arv2 (p)
     Se (p == Nulo)
     q = p;   qant = Nulo
     Enquanto ( q != Nulo )
          Se (qant == q->pai)                                    // Caso 1
               qant = q
               Se (q->esq != Nulo)    q = q->esq
               Sse (q->dir != Nulo)   q = q->dir
               Senão                     calcAlt(q);   q = q->pai
          Se (qant == q->esq)                                    // Caso 2
               qant = q
               Se (q->dir != Nulo)    q = q->dir
               Senão                     calcAlt(q);   q = q->pai
          Senão                                                  // Caso 3
               qant = q
               calcAlt(q);   q = q->pai
```

◇