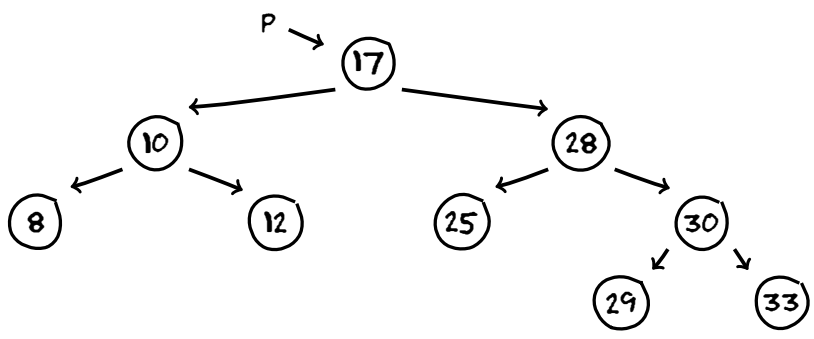
A operação de remoção

Imagine que o ponteiro `p` aponta para uma árvore binária de busca

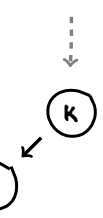


A tarefa é

→ *Remover o elemento* `k` *da árvore*

Bom, aqui nós temos dois casos.

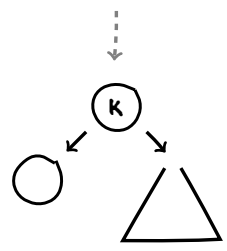
No primeiro caso, o elemento `k` não possui o filho direito



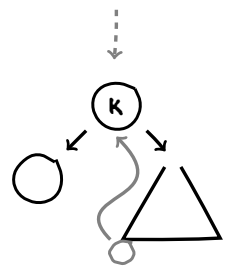
E aqui é bem fácil ver que a remoção do `k` é semelhante à remoção do menor elemento

— (*i.e., o seu filho esquerdo fica no seu lugar*)

No segundo caso, existe uma subárvore à direita do `k`



E aqui, a esperteza é remover o menor elemento da subárvore direita, para colocar no lugar do `k`



A coisa fica assim

```
func remoçãoArv-Rec (q,k)

      Se (q == Nulo)   Retorna (Nulo,Nulo)              // o k não está aqui ...

      Sse (q->num > k)
            fi,r = remoçãoArv-Rec (q->esq,k)            // descida na árvore
            q->esq = fi                                 // reconexão
            Retorna (q,r)

      Sse (q->num < k)
            fi,r = remoçãoArv-Rec (q->dir,k)            // descida na árvore
            q->dir = fi                                 // reconexão
            Retorna (fd,q)



    Senão
          Se (q->dir == Nulo)   Retorna (Nulo,Nulo)
                f = q->esq;   q->esq = Nulo
                Retorna (f,q)
          Senão
                f,menor = RMA-Rec (q->dir)
                menor->esq,dir = q->esq,f
                q->esq,dir = Nulo,Nulo
                Retorna (menor,q)
```

◇