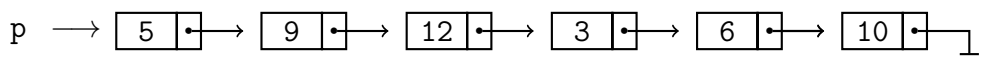
Imagine que o ponteiro p aponta para uma lista encadeada



A tarefa é

→ *Mover o maior elemento para o fim da lista*

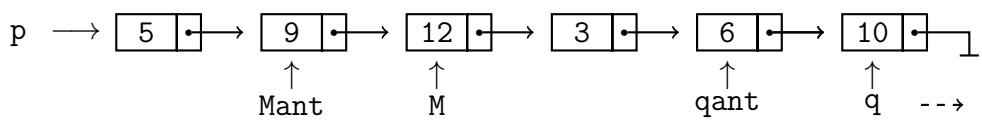
Bom, aqui a gente precisa fazer a coisa em etapas

1. primeiro, a gente encontra o maior elemento
2. daí, a gente remove esse elemento da lista
3. e daí, o maior elemento é recolocado no final da lista

Certo.

Para encontrar o maior, a gente vai utilizar a técnica de percorrer a lista com dois ponteiros.

E daí, sempre que o ponteiro para o maior é atualizado, nós também atualizamos o ponteiro que aponta para o elemento que fica antes do maior.



A coisa fica assim

```
func maiorFim ( p )

      Se ( p == Nulo  OU  p->prox == Nulo)   Retorna (p)

      M = p;    Mant = Nulo
                                                    // o maior até aqui
      q = p-prox;    qant = p
      Enquanto (q != Nulo )                         // procurando o maior
            Se ( q->num > M->num )
                  M= q;    Mant = qant
            qant = q;    q = q->prox

      Se ( M->prox == Nulo )   Retorna (p)          // o maior já é o último

      Se ( Mant == Nulo )    p = M->prox            // o maior é o 1o elemento
      Senão                  Mant-prox = M->prox
      M->prox = Nulo

      qant->prox = M                                // 3a etapa
      Retorna (p)
```

Nota: Depois que a gente termina de percorrer a lista (1a etapa), o ponteiro `qant` fica posicionado no último elemento. E é por isso que a 3a etapa fica tão simples.

◇