Dividir listas encadeadas

Considere uma lista encadeada com nó cabeça `le` definida por células

```
typedef struct celula {
   int dado;
   struct celula *prox;
} celula;
```

Faça uma função

```
void divide_lista (celula *l, celula *l1, celula *l2);
```

que recebe uma lista encadeada encabeçada por `l` e a divide em duas listas `l1` e `l2` de forma que `l1` contenha todos os *numeros ímpares* de `l` (na ordem em que aparecem em `l`) e `l2` todos os *números pares* de `l` (na ordem em que aparecem em `l`).

Observações

1. Você não deve alocar nenhuma nova célula na sua função, apenas manipular os ponteiros dos nós de `l` para que estejam em `l1` ou `l2`.
2. Você deve considerar que os nós cabeça `l1` e `l2` já foram alocados antes da chamada para a função `divide_lista`.
3. Como consequência, a lista encabeçada por `l` não estará intacta após a chamada à sua função.

Exemplo

Suponha, por exemplo, que a lista `l` seja

`l -> 10 -> 4 -> -9 -> 2 -> 7 -> 10 -> NULL`

Sua função deve devolver as listas

`l1 -> -9 -> 7 -> NULL`

e

`l2 -> 10 -> 4 -> 2 -> 10 -> NULL`

Author: John L. Gardenghi