Lista de Exercícios

Para as questões de lista ligada considere as estruturas a seguir:

```
struct LL_Int{
  int valor;
  struct LL_Int *proximo;
};

struct LDL_Int{
  int valor;
  struct LDL_Int *proximo;
};

struct ListaDuplaInt{
  LDL_Int *primeiro;
  LDL_Int *ultimo;
};
```

1. Escreva o código de uma função que insira um elemento no início da lista ligada. A função deve retornar um ponteiro para o novo início da lista. Determine a complexidade da sua função usando a notação big-Oh.

   ```
   struct LL_Int * inserir_inicio(struct LL_Int *inicio, int valor);
   ```

2. Escreva o código da função a seguir, que retorna se um valor v está na lista cujo início é passado como parâmetro. Defina a complexidade usando a notação big-Oh.

   ```
   int buscar(struct LL_Int *inicio, int v);
   ```

3. Escreva o código de uma função que mova o maior elemento para o final da lista. A função deve retornar um ponteiro para o início da lista. Defina a complexidade usando a notação big-Oh.

   ```
   struct LL_Int * mover_maior_fim(struct LL_Int *inicio);
   ```

4. Escreva uma função que receba como parâmetro o ponteiro para o início de uma lista ligada e retorne a quantidade de elementos na lista. Determine a complexidade da sua função usando a notação big-Oh.

   ```
   int quantidade(struct LL_Int *inicio);
   ```

5. Escreva o código de uma função que remova o último elemento de uma lista duplamente encadeada. A função deve retorna 0 se não houve remoção ou 1 se o último tiver sido removido. Determine a complexidade da sua função usando a notação big-Oh.

   ```
   int remover_ultimo(struct ListaDuplaInt *lista);
   ```

6. Escreva o código de uma função para contar a quantidade de vezes que um valor v ocorre em uma lista duplamente ligada. Determine a complexidade da sua função usando a notação big-Oh.

   ```
   int ocorrencias(struct ListaDuplaInt *lista, int v);
   ```