Exercícios - Lista Simplesmente Encadeada com Descritor

1. Implemente uma função para imprimir o descritor de uma lista:
    a. A função deve imprimir:
        - Valor do nó inicial
        - Valor do nó final
        - Tamanho da lista
    b. Caso o ponteiro para o nó inicial ou para o nó final seja nulo, deve-se imprimir “NULL”.
    c. Exemplo:

```
Lista *lista = new Lista();
insereInicioL(lista, 10);
insereInicioL(lista, 20);
insereInicioL(lista, 30);
insereInicioL(lista, 50);
mostraDescritorL(lista);
cout << "L[" << lista->tamanho << "]:{" << lista << "}" << endl;
delete(lista);
```

```
Descritor [4, 50, 10]
L[4]:{50, 30, 20, 10}
```

2. A empresa XYZ possui uma lista com as informações de todos os funcionários. Para cada funcionário, as informações de código, nome, idade e salário são registradas. Com base nessas informações, crie um programa em C++ para gerenciar a lista de funcionários como segue:
    a. Implemente uma lista simplesmente encadeada, projetada para armazenar os dados dos funcionários da empresa.
    b. Crie um descritor para a lista contendo:
        - Ponteiro para o primeiro nó da lista
        - Ponteiro para o último nó da lista
        - Quantidade de nós
    c. Implemente as seguintes funções:
        - Função para inserir funcionários no início da lista
        - Função para inserir funcionários no final da lista
            - Utilize o descritor para o último nó da lista, ou seja, não é necessário percorrer a lista para realizar a inserção.
        - Função para imprimir os nós da lista:
            - Imprimir os dados de cada nó (código, nome, idade e salário).
            - Implemente via sobrecarga do operador <<, de forma que seja possível imprimir a lista com o comando *cout*.
        - Função para imprimir a média dos salários pagos pela empresa.
        - Função para imprimir o nome de todos os funcionários que possuem salário menor que a média.
        - Função para ordenar a lista de funcionários em ordem alfabética:
            - **Não é permitido** o uso de vetores ou estruturas de dados auxiliares. Altere apenas os valores dos nós, com base na comparação de *strings*. Não há necessidade de alterar os valores dos ponteiros, somente a variável *dado*.
            - Use a função **stccmpi** da biblioteca **<cstring>** para comparar duas *strings*. Exemplo:

```cpp
//strcmpi -> ordem lexicográfica (ignora Maiúsculas e Minúsculas)

string a("abobora"), b("Abobora");

if( strcmpi(a.c_str(), b.c_str()) == 0 )
    cout << "A=B" << endl; //vai imprimir esta linha
else if( strcmpi(a.c_str(), b.c_str()) < 0 )
    cout << "A<B" << endl;
else if( strcmpi(a.c_str(), b.c_str()) > 0 )
    cout << "A>B" << endl;

a= "Ana";
b= "Fernando";
cout << "strcmpi(" <<a << ", " << b << ")=" << strcmpi(a.c_str(), b.c_str()) << endl; //vai imprimir -5

a= "Ana";
b= "Adão";
cout << "strcmpi(" <<a << ", " << b << ")=" << strcmpi(a.c_str(), b.c_str()) << endl; //vai imprimir 10
                                                int strcmpi(const char* _Str1, const char* _Str2)
a= "João";
b= "Joana";
cout << "strcmpi(" <<a << ", " << b << ")=" << strcmpi(a.c_str(), b.c_str()) << endl; //vai imprimir 130

a= "Gabriela";
b= "Pedro";
cout << "strcmpi(" <<a << ", " << b << ")=" << strcmpi(a.c_str(), b.c_str()) << endl; //vai imprimir -9
```