Instituto Federal de Sergipe
Bacharelado em Sistemas de Informação
Disciplina: Estrutura de Dados I **Semestre:** 2018.1
Professor: Francisco Rodrigues Santos

— Lista Sequencias e Encadeadas —

Nome: _____

- É permitido o uso de códigos próprios.
- Não será permitido o uso de coleções implementadas pelo Java, como ArrayList.

1. Implemente a interface ILista a seguir que representa uma lista não ordenada.

```java
interface ILista{
    /**
    * Inicializa a lista não ordenada informando a quantidade máxima.
    * @param quantidadeMaxima
    */
    void inicializar(int quantidadeMaxima);
    /**
    * Adiciona um elemento na lista
    * @param o Elemento a ser adicionado
    * @throws ListaException Erro caso não tenha mais espaço disponível.
    */
    void adicionar(Comparable o) throws ListaException;
    /**
    * Remove um item da lista
    * @param chave informar a chave de busca do item
    * @throws ListaException Erro caso não tenha o item informado.
    */
    void remover(Object chave) throws ListaException;
    /**
    * Caso a chave seja encontrada, retorna verdadeiro
    * @param chave
    * @return
    */
    boolean contem(Object chave);
}
```

Lista não ordenada

```java
class ListaException extends Exception {
    public ListaException(String msg){
        super(msg);
    }
}
```

Exemplo de classe de erro. Personalize caso necessário.

2. Adicione a funcionalidade para obter um determinado item a partir de sua chave.

```java
/**
* Recupera o item da lista. Caso não encontre, retorna null.
* @param chave a chave de busca do item
* @return Caso não encontre, retorna null.
*/
Comparable obter(Object chave);
```

3. Adicione a funcionalidade para obter um determinado item a partir de sua posição na lista.

```java
/**
* Recupera o item da lista com base em sua posição.
* @param chave a chave de busca do item
* @return O item da posição informada
* @throws ListaException Caso a posição seja inválida.
*/
Comparable obter(int posicao) throws ListaException;
```

4. Adicione a funcionalidade para ordenar os elementos na lista.

```
/**
* Ordena os itens da lista
*/
void ordenar();
```

5. Adicione a funcionalidade aumentar a quantidade disponível em X porcento.

```
/**
* Aumenta a quantidade de itens disponíveis na lista em X porcento.
* @param percentual percentual a ser aumentado.
*/
void expandirLista(float percentual);
```

6. Juntar a sub-lista adicionando os elementos no final

```
    /**
    * Adiciona a outra lista no final da lista.
    * @param outralista a outra lista a ser somada.
    */
    void UnificarNoFinal(IList outralista);
```

7. Juntar a sub-lista adicionando os elementos no meio

```
    /**
    * Adiciona a outra lista após o item chave existente na lista.
    * @param outralista a outra lista a ser somada.
    * @param chave elemento chave que antecederá a outra lista
    */
    void UnificarAposOItemChave(IList outralista, Object chave);
```