**Guia ArrayList**

**Vamos trabalhar com Generics<>**

**Tags:**

List<elemento desejado> atributo –

**Elemento.add(new classe(parâmetro do método))** –

public void adicionarTarefa(String descricao){  
 tarefaList.add(new Tarefa(descricao));  
}

adiciona um elemento dentro da lista pode ser adicionado a tarefa ou pelo índice(int), tarefa(element)

**Elemento.addAll()** – adiciona todos os elementos, ou coleção

Elemento.remove() – temos os seguintes de parâmetros para esse remove, temos remove(Object o), remove(int index), removeAll(Collectionm<?> c) e o removeIf(Predicate<? Super elemento> filter)

Abaixo está sendo usado o removeAll:

public void removerTarefa(String descricao) {  
 List<Tarefa> tarefaParaRemover = new ArrayList<>();  
 for (Tarefa t: tarefaList) {  
 if(t.getDescricao().equalsIgnoreCase(descricao)){  
 tarefaParaRemover.add(t);  
 }  
 }  
 tarefaList.removeAll(tarefaParaRemover);  
}

**Criando um construtor:** Que sempre que chamarmos um novo objeto já for criado uma **List Vazia**

Nesse caso abaixo o construtor ListaTarefa ira toda vez que for chamada uma lista vazia (tarefaList) com isso conseguimos adicionar elementos, remover caso já tenha elementos na lista, obter a quantidade de elemento na Lista etc

public ListaTarefa() {  
 this.tarefaList = new ArrayList<>();  
}

No caso abaixo temos a classe Tarefa que vai representar o elemento que vai dentro da List<Tarefa>

private List<Tarefa> tarefaList;

elemento.size() - Retorna o número total de elemento que tem dentro da lista

public int obterNumeroTotalTarefas() {  
 return tarefaList.size();  
}