Exercícios com Collections – parte 2.

1) Utilize o mesmo projeto do exercício anterior.
2) No pacote de testes, crie a classe TestaCollection5 conforme abaixo.

```
12 public class TestaCollection5 {
13     public static void main(String[] args) {
14         List<Conta> lista1 = new ArrayList();
15         ContaCorrente cc = new ContaCorrente(2);
16         cc.deposita(5000);cc.getTitular().setNome("Rafael");
17         ContaPoupanca cp = new ContaPoupanca(1);
18         cp.deposita(2500);cp.getTitular().setNome("Maria");
19         ContaCorrente cc2 = new ContaCorrente(5);
20         cc2.deposita(3000);cc2.getTitular().setNome("Fulano");
21 
22         lista1.add(cc);lista1.add(cp);lista1.add(cc2);
23 
24         for (int i = 0; i < lista1.size(); i++) {
25             Conta conta = lista1.get(i);
26             System.out.println("Conta nº "+conta.getNumero()
27             +" com saldo de R$"+conta.getSaldo()+" - Titular: "+conta.getTitular().getNome());
28         }
29 
30         System.out.println();
31         //Como ordenar Contas utilizando o 2º argumento de sort???
32         //Collections.sort(lista1, new ContaComparatorSaldo());
33         //Collections.sort(lista1, new ContaComparatorNome());
34 
35         for (Conta conta : lista1) {
36             System.out.println("Conta nº "+conta.getNumero()
37             +" com saldo de R$"+conta.getSaldo()+" - Titular: "+conta.getTitular().getNome());
38         }
39     }
40 }
```

3) Crie a classe ContaComparatorSaldo mencionada na linha 32 conforme código abaixo

```
7  public class ContaComparatorSaldo implements Comparator<Conta> {
8  
9      @Override
10     public int compare(Conta c1, Conta c2) {
11         return (int) (c1.getSaldo()-c2.getSaldo());
12     }
13 
14 }
```

4) Descomente a linha 32 e veja o resultado
5) Crie a classe ContaComparatorNome mencionada na linha 33 conforme código abaixo

```
5  public class ContaComparatorNome implements Comparator<Conta>{
6  
7      @Override
8      public int compare(Conta c1, Conta c2) {
9          return c1.getTitular().getNome().compareTo(c2.getTitular().getNome());
10     }
11 
12 }
```

Perceba que não há como perguntar se um nome é maior que o outro. No entanto, o nome do titular é uma String e a classe String já sabe se ordenar. Logo, basta invocar o compareTo a partir da 1ª String passando a 2ª String como argumento!!

6) Comente a linha 32, descomente a linha 33 e veja o resultado