Programação III

Semestre de Inverno de 2020-2021

2º Trabalho prático

Data de Entrega: 8 de Janeiro de 2022

Grupo 2

1. Realize na classe **`AlgorithmUtils`** o método estático público com a seguinte assinatura:

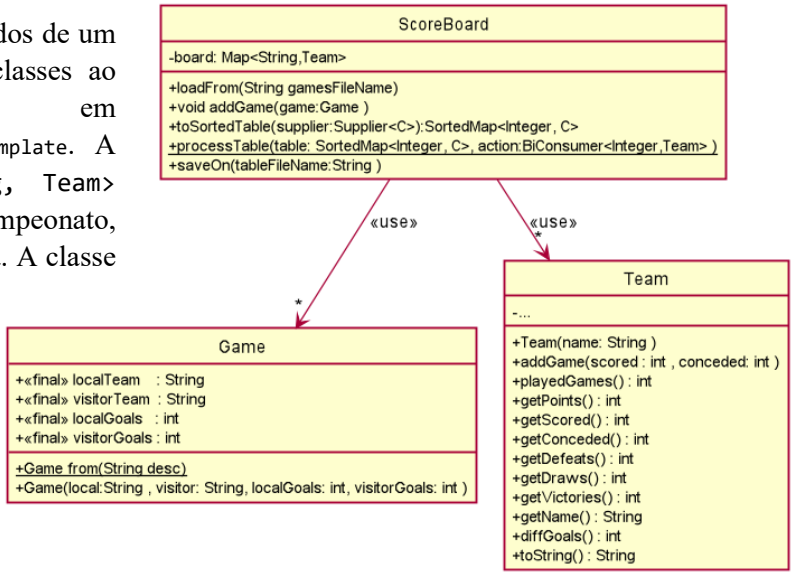
```
<V,C extends Collection<V>> List<C>

getSubsequences( Iterable<V> sequence,
                 Comparator<V> compareValue,
                 Supplier<C> supplierList)
```

que produz uma lista de subsequências ordenadas, de forma crescente segundo o comparador **`cmp`**, existentes na sequência **`sequence`**. Exemplo com uma sequência de inteiros:

Sendo a sequência => [10, 20, 30, 12, 13, 8, 1, 2, 3]

Produz a lista de subsequências => [[10, 20, 30], [12, 13], [8], [1, 2, 3]]

2. De modo a realizar estatísticas sobre os resultados de um campeonato de futebol, foram definidas as classes ao lado, implementadas parcialmente em `https://github.com/isel-leirt-leetc-pg3/trab2_template`. A classe `ScoreBoard` mantém um `Map<String, Team>` com as estatísticas das equipas(`Team`) do campeonato, usando como chave o nome da respetiva equipa. A classe `Game` representa o resultado de um jogo.



   a) Complete os métodos `addGame`, `getPoints` e `playedGames` em falta na classe `Team`.

   b) Acrescente à classe `Game` o método estático `Game from(String desc)` que cria uma instância de `Game` a partir da *string* com o formato:

`<local_name> : <local_score> - <visitor_name> : <visitor_score>` (ex: `Benfica:3 – Porto:1`)

   c) Realize os métodos em falta na classe `ScoreBoard`:

      i. O método `addGame(Game game)` da classe `ScoreBoard` adiciona o resultado de `game` ao mapa do *score board*.

      ii. O método `loadFrom(String gamesFileName)` adiciona ao mapa do *score board* os resultados presentes no ficheiro de nome `gamesFileName`. Note que o método `loadFrom` pode lançar a excepção `IOException`. Os resultados existentes no ficheiro seguem o formato apresentado na alínea anterior.

      iii. O método genérico

```
public <C extends Collection<Team>>
SortedMap<Integer, C> toSortedTable(Supplier<C> supplier)
```

produz, a partir dos pares presentes no mapa `board`, outro mapa ordenado pela chave inteira que representa uma dada pontuação, e cujo valor é a coleção das equipas com essa pontuação. Cada coleção é obtida do fornecedor `supplier`.

iv. O método genérico estático

```
public static  <C extends Collection<Team>>
void processTable(SortedMap<Integer, C> table, BiConsumer<Integer,Team> action)
```

recebe o mapa `table`, em que a chave são os pontos e o valor associado as equipas com esses pontos, executandp a ação `action` por cada par (pontos, equipa) disponível em `table`.

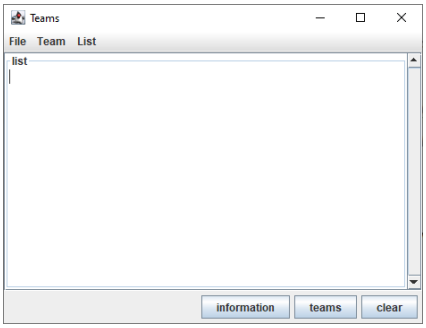
v. O método

```
saveOn(String tableFileName)
```

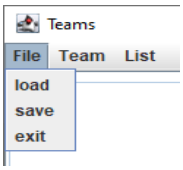
cria o ficheiro de texto de nome `tableFileName` com a tabela classificativa do campeonato, ordenando as equipas por ordem decrescente dos seus pontos. As equipas com os mesmos pontos devem ser apresentadas por ordem decrescente da sua diferença de golos. Apresenta-se abaixo um exemplo do formato pretendido.

```
Name          Points  Games   Victories Draws  Defeats  Scored  Conceded
Benfica          4       2       1       1       0       5       3
Porto            4       2       1       1       0       4       3
Braga            3       2       1       0       1       4       4
Sporting         3       2       1       0       2       3       6
```

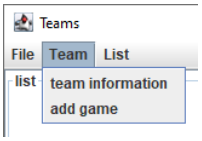
3. Realize a aplicação com interface gráfica **`TeamsStats`**, e aspeto semelhante ao das figuras abaixo, que tira partido das classes desenvolvidas na questão anterior para produzir e visualizar informação sobre as equipas. Pretende-se funcionalidades para:



- Ler jogos de ficheiros e escrever as pontuações em ficheiros.



- Adicionar um jogo e mostrar a informação de uma determinada equipa.



- Listar informação sobre as equipas: nomes das equipas em jogo, pontuações, e equipas e respetiva pontuação.

Bom trabalho