INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PARAÍBA Campus Campina Grande	**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA – CAMPUS CAMPINA GRANDE**		
	CURSO:	**CURSO SUPERIOR ENGENHARIA DA COMPUTAÇÃO**	
	PERÍODO:		**TURMA:**
	DISCIPLINA:	**PROGRAMAÇÃO ORIENTADA A OBJETOS**	
	PROFESSOR:	**CÉSAR ROCHA VASCONCELOS**	SEMESTRE LETIVO

NOME:	NOTA:
	DATA: / /

PRÁTICA STREAMS & COLEÇÕES

1. Crie um projeto no Maven chamado **proj-cities** e importe-o na sua IDE de preferência (IntelliJ ou Eclipse, por exemplo).
2. Primeiro, no UML, modele a classe chamada **`CitiesProcessor`**. Esta classe deve ter os seguintes atributos e métodos:
   a. **`private Set<City> citiesSet`**, Este atributo deve armazenar um conjunto de objetos de uma classe `City` (cidade);
   b. **`public Set<City> buildSetOfCities( Path filePath )`**, capaz de ler os dados de um arquivo de cidades (`cities_duplicatas.txt`), remover eventuais duplicatas existentes, montar e retornar um conjunto (e.g., `TreeSet<City>`) contendo objetos da classe `City`. Estude o arquivo `cities_duplicatas.txt` e modele no UML uma segunda classe `City`, com seus atributos, construtores e outros métodos que você julgar necessários; Agora, verifique qual o relacionamento é o mais adequado no UML para relacionar a classe `City` à `CitiesProcessor`.
   c. No UML da classe `City`, inclua os métodos `equals`, `hashCode` e `compareTo` (cuja separação de duplicatas e ordenação padrão devem ser feitas usando o nome da cidade). Por conta de `compareTo`, verifique qual o relacionamento é o mais adequado no UML para relacionar a classe `City` à interface `Comparable<City>`.
   d. No UML da classe `CitiesProcessor`, modele agora o método **`public void writeSetOfCities( Path pathDestino, Comparator<City> comparator )`**. Este método deve ser capaz de transferir todos os objetos `City` do atributo **`citiesSet`** para um arquivo texto a ser gravado em um **`pathDestino`** no disco, obedecendo a ordem definida por um comparador (`Comparator<City>`). Ao gerar o arquivo texto, o método **`writeSetOfCities`** deve organizar sempre uma cidade por linha.
   e. Ainda no UML, modele dois comparadores: `ComparatorByZipCode` e `ComparatorByState`. Estas classes serão os comparadores usados pelo método **`writeSetOfCities`** para ordenar as cidades pelo código-postal e estado, respectivamente. Agora, verifique qual o relacionamento é o mais adequado no UML para relacionar cada uma destas classes comparadoras acima à interface `Comparator<City>`. Também verifique qual é o relacionamento mais adequado no UML entre os comparadores e `CitiesProcessor`.
   f. Ainda no UML, modele uma outra classe `CitiesProcessorDemo` que possui apenas o *main*(), para executar e verificar o funcionamento da classe `CitiesProcessor`. Verifique qual o relacionamento é o mais adequado no UML para relacionar a classe `CitiesProcessorDemo` à `CitiesProcessor`. Um exemplo de uso desta classe é fornecido no quadro abaixo.
   g. No `pom.xml`, indique qual das classes possui o *main* e será usada na produção do arquivo executável JAR da aplicação.
   h. Rode o arquivo JAR executável gerado pelo Maven. Sua aplicação deve funcionar normalmente e os 3 arquivos-texto devem ser produzidos sem duplicatas ou erros na ordenação.

```
public static void main( String[] args ) {
  CitiesProcessor cp = new CitiesProcessor(); // criar o processador de cidades

  // ler o arquivo texto e montar o set de objetos-cidades
  Set<City> set = cp.buildSetOfCities( Path.of( "files/cities_duplicatas.txt" ) );
  cp.setCitiesSet( set ); // passar o set montado para o objeto CitiesProcessor

  // gerar e salvar um arquivo texto com as cidades ordenadas pelo nome (o default)
  cp.writeSetOfCities( Path.of( "files/cities_ordered_by_name.txt" ), Comparator.naturalOrder() );

  // gerar e salvar um outro arquivo texto com as cidades ordenadas pelo zip code
  ComparatorByZipCode comparadorZip = new ComparatorByZipCode();
  cp.writeSetOfCities( Path.of( "files/cities_ordered_by_zipcode.txt" ), comparadorZip );

  // gerar e salvar arquivo texto com as cidades agora ordenadas pelo estado
  ComparatorByState comparadorState = new ComparatorByState();
  cp.writeSetOfCities( Path.of( "files/cities_ordered_by_state.txt" ), comparadorState );

} // main
```