## **Exercise: Functional Interface and Lambda Expressions**

**Requirement: (Basic lambdas)**

Consider a list containing few strings like

new String[] {"Ryan", "Alfred", "Beth", "Emma", "Bella", "Edward"}

and sort it by -

1. length (i.e., shortest to longest)
2. reverse length (i.e., longest to shortest)
3. alphabetically by the first character only (Hint: charAt(0) returns the numeric code for the first character)
4. Strings that contain “e” first, everything else second. For now, put the code directly in the lambda.

**Requirement: (Static helper method)**

Redo the previous problem, but use a static helper method so that your lambda looks like this:

Arrays.sort(words, (s1, s2) -> Utils.yourMethod(s1, s2))

Consider a Person entity:

public class Person {

private String name;

private int age;

}

**Requirement: (Write a lambda function)**

Given a list of People, filter all people with names starting with 'a' or 'm'

**Requirement: (Write a lambda function)**

Given a list of People, filter all people with gender “Male” and age greater than 20.

**Generalizing Filter: (Make use of functional interface and lambda expressions)**

The **TransactionPredicate** and **PersonPredicate** provide a contract that has the same meaning.

***Take an entity, and return a boolean value.***

Write a common Predicate that can be used to for both Person as well as Transaction Predicates. *(Hint : Use Generic Type to represent the accepted entity in the filter method and annotate it as functional interface and then using lambda expression use it inside code to fulfil requirements)*

**Additional Tasks:**

* **The @FunctionalInterface annotation**. Add the annotation to your interfaces. Does adding this annotation change the behavior of your code? What happens if you try to add a second abstract method to functional interface?
* Find out common functional interfaces in Java 8.