Exercises: Lambda Expressions Part 2

These exercises are dramatically easier than the previous set. Start by copying and renaming your project from the previous set of exercises.

1. **The @FunctionalInterface annotation.** Add the annotation to your two interfaces (TwoString-Predicate and TwoElementPredicate). Does adding this annotation change the behavior of your code? What happens if you try to add a second abstract method to TwoStringPredicate?

2. **Method references.** On the previous set of exercises, your solutions to the sorting problems looked something like this:

   - Bullet 1 Arrays.sort(words, (s1,s2) -> someValue);
   - Bullet 2 Arrays.sort(words, (s1,s2) -> someValue);
   - Bullet 3 Arrays.sort(words, (s1,s2) -> someValue);
   - Bullet 4 Arrays.sort(words, (s1,s2) -> { some code;
some more code;
even more code;
return(someValue); }
   - Bullet 5 Arrays.sort(words, (s1,s2) -> Utils.yourMethod(s1,s2))

   For that very last example (bullet 5), replace the explicit lambda with a method reference.

3. **More method references.** Following is some imaginary code; you can get this code by copying method-references.txt from the lambdas-2-exercises project. Change each to use method references instead of explicit lambdas. Hint: you can do this without knowing what any of the code does.

```
method1(x, y, d -> Math.cos(d));
someList.forEach(entry -> System.out.println(entry));
method2(a, b, c, (d1,d2) -> Math.pow(d1,d2));
someStream.reduce(0, (i1,i2) -> Integer.sum(i1, i2));
method3(foo, bar, (a,b,c) -> Utils.doSomethingWith(a,b,c));
method4(() -> Math.random());
```