# Lambda Expressions: Part 3

Assuming that you use varargs in your solutions to at least some the problems below, note that you will receive an odd-sounding warning (not error) about potential heap pollution. It is safe to ignore this error for now, but in the file IO lecture we will briefly explain the warning and show how to suppress it with @SafeVarargs.

**1.** Make a method called allPassPredicate that accepts any number of generically typed Predicates, and returns a single Predicate that tests if the argument passes all of the input Predicates. Make a method called firstAllMatch that takes a Stream and any number of correspondingly-typed Predicates, and returns the first entry that matches all of the Predicates. For example, if words is a List<String>, the following would find the first word that *both* contains an "o" *and* has length greater than 5.

```
FunctionUtils.firstAllMatch(words.stream(),
                            s -> s.contains("o"),
                            s -> s.length() > 5);
```

**2.** Make a method called anyPassPredicate that accepts any number of generically typed Predicates, and returns a single Predicate that tests if the argument passes any of the input Predicates. Make a method called firstAnyMatch that takes a Stream and any number of correspondingly-typed Predicates, and returns the first entry that matches any of the Predicates. For example, if words is a List<String>, the following would find the first word that *either* contains an "o" *or* has length greater than 5.

```
FunctionUtils.firstAnyMatch(words.stream(),
                            s -> s.contains("o"),
                            s -> s.length() > 5);
```