Concurrent and Parallel Programming in Java 8

part II – Lambda functions

1. The following sentence contains an error. Please indicate it and propose a solution:

```
System.out.println((x -> x + 1).apply(1));
```

2. Define the predicates `even(x)` and `odd(x)`, which takes a single integer argument `x` and returns a `Boolean` value that indicates if the argument is even or odd. Please define one of the predicates in terms of the other.
3. Given the following class definition

```
class MyTest {
   List<String> arr = Arrays.asList("Mariapia", "Teresa", "Stefano");
}
```

   Define the method `private String filter(Predicate<String> p)`, which returns the concatenation of all the strings in the array which fullfills the condition `p`.
   Define the method `private String update(Function<String, String> f)`, which returns the concatenation of all strings in `arr` modified by applying the function `f`. Show examples about its use with predefined methods of the class `String` passed by using method references.
4. The method `sort()` for collections uses a comparator to define the order relation. How do you think a lambda expression can be used for that purpose? Evaluate yourself your solution with an example, which you will present next class.
5. Define an example that shows that lambda examples are evaluates only when required (laziness).
6. Analyze the following piece of code:

```
BiFunction<Integer, Integer, Integer> f1 = (x, y) -> x + y;
Function<Integer, Function<Integer, Integer>> f2 = x -> y ->
f1.apply(x, y);
System.out.println(f1.apply(2, 3)); // output is 5
System.out.println(f2.apply(2).apply(3)); // output is 5
```

   It uses a concept called »currying«. Please investigate it and prepare a small example of »currying« to be discussed next class.