CS2030 Programming Methodology
Semester 2 2021/2022

23 & 23 March 2022
Problem Set #8

1. Write a method `fib(int a, int b)` that returns an `InfiniteList<Integer>` where the elements of the infinite list are the Fibonacci numbers starting from `a` and `b`.

   ```
   fib(1,1).head(); // returns 1
   fib(1,1).tail().head(); // returns 1
   fib(1,1).tail().tail().head(); // returns 2
   fib(1,1).tail().tail().tail().head(); // returns 3
   ```

   Next, write another method that returns the n-th Fibonacci number using your `fib` method.

   Now, write a method that returns the first n Fibonacci numbers as a `Stream<Integer>`.

   For instance, the first 10 Fibonacci numbers are $1, 1, 2, 3, 5, 8, 13, 21, 34, 55$.

   Hint: Write an additional `Pair` class that keeps two items around in the stream

2. `IntStream` is the `int` primitive version of `Stream`. Write a method `omega` with signature `IntStream omega(int n)` that takes in an `int n` and returns a `LongStream` containing the first n omega numbers.

   The i^{th} omega number is the number of distinct prime factors for the number i. The first 10 omega numbers are $0, 1, 1, 1, 1, 2, 1, 1, 1, 2$.

   The `isPrime` method is given below:

   ```
   boolean isPrime(int n) {
       return IntStream
           .range(2, n)
           .noneMatch(x -> n%x == 0);
   }
   ```

3. Write a method product that takes in two `List` objects `list1` and `list2`, and produce a `Stream` containing elements combining each element from `list1` with every element from `list2` using a `BiFunction`. This operation is similar to a Cartesian product.

   ```
   public static <T,U,R> Stream<R> product(List<? extends T> list1,
           List<? extends U> list2,
           BiFunction<? super T, ? super U, R> func)
   ```

   For example, the following program fragment:

```
List<Integer> list1 = List.of(1,2,3,4);
List<String> list2 = List.of("A","B");
product(list1, list2, (str1, str2) -> str1 + str2)
    .reduce("", (x, y) -> x + y + " ")
```

gives the output:

```
1A 1B 2A 2B 3A 3B 4A 4B
```