The purpose of this assignment is to learn how to use Streams in Java.

Exercise 1 – Population

Import the source code contained in *es1* in your IDE. The program creates a list of random people (Students, Workers and Person) and outputs statistics about the population. To compute the statistics, utility methods accepting a list of Person instances and lambda expressions are used. The goal of this exercise is to remove those utility methods and all explicit iterations over the collections and replace them with stream operations.

1) Remove for loops in:
   a) *init* method by using the *Stream.generate* method in combination with the *limit*, *peek* and *collect* methods to create and print the initial list of Person instances.
   b) *search* method by requesting a stream from the passed population parameter and using the *filter* and *collect* methods to return a new collection of Person instances.
   c) *categorizeString* / *categorizeInteger* methods, by requesting a stream from the passed population parameter and using *collect* and *Collectors.groupingBy* methods to create a new map.
   d) *printCategorizedString* / *printCategorizedInteger* methods, by requesting a stream of *Entry* from the passed *categorized* map's entrySet and use the forEach method on the stream.
   e) *printPersonList* method, by requesting a stream from the passed people list and then using *map, collect and Collectors.joining* methods to generate the string to be printed. The map operation should be used to transform a Person instance to a String, using the *toString* method.
2) Remove the *CategorizeOperation* / *EvaluateOperation* interfaces and the *search* / *categorizeString* / *categorizeInteger* helper methods by moving the method's logic to the main method where the methods are invoked, considering that:
   a) *CategorizeOperation* interface is like the *Function* interface used by the *Collectors.groupingBy* method.
   b) *EvaluateOperation* interface is like the *Predicate* functional interface used by the *Stream.filter* method.

Exercise 2 – Text analyzer

Write a program that reads a text file and outputs the following information, without using any for / while loops but instead using only streams:

- the longest word
- the shortest word with at least 5 characters
- the word that contains the highest number of vowels
- the list of words that start with a vowel, without duplicates
- the list of words that start with a capital T, without duplicates
- the number of words that start with a vowel, and the number of remaining words (hint: create 2 groups and print their size)
- general statistics about the word lengths such as: sum of lengths, min, average and max length.

HINTS:

- The following one-liner may be used to read the contents of a text file into "words":

```
Arrays.asList(new String(Files.readAllBytes(Paths.get("/path/to/file.txt"))).split("\\s+"))
```

Exercise 3 – Generators (Optional)

Write a program that generates the following Streams:

- Even numbers using both *IntStream.iterate* and *Stream.iterate* methods
- Card symbols (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K) by using:
  - *Stream.of* method

  - *IntStream.closedRange* method in combination with a map function that converts the int value to the corresponding char/String representing the card's symbol.
- A 18-character long password by generating a stream of random numbers in the ASCII range (40 - 120) and converting them to chars / Strings.
  Finally create a String from the character stream by using:
  - Collectors.joining
  - Stream.reduce
- The Fibonacci sequence (0, 1, 1, 2, 3, 5, 8, …) using the *LongStream.generate* method.
  Hint: create a helper class to store the two values (n1, n2) used to compute the sequence and a *next* method that returns the current value and updates n1 and n2 to compute the next Fibonacci value. Use the implemented *next* method in combination with the generate method.
  $F_n = F_{n-1} + F_{n-2}$;

  $F_0 = 0$
  $F_1 = 1$
  $F_2 = F_1 + F_0 = 1 + 0 = 1$
  $F_3 = F_2 + F_1 = 1 + 1 = 2$
  $F_4 = F_3 + F_2 = 2 + 1 = 3$
  $F_5 = F_4 + F_3 = 3 + 2 = 5$