Programming languages Java ZH 2022.11.26. practice

Due No due date **Points** 20 **Questions** 1
Available Nov 26 at 10:40am - Nov 26 at 1pm about 2 hours **Time Limit** None

This quiz was locked Nov 26 at 1pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	134 minutes	0 out of 20 *

* Some questions not yet graded

Correct answers are hidden.

Score for this quiz: **0** out of 20 *
Submitted Nov 26 at 12:58pm
This attempt took 134 minutes.

Question 1 **Not yet graded / 20 pts**

Programming Languages Java exam, practical part

Conditions

Do the following **right now**: make sure that no communication device is available to you.

- **Put away** phones, headphones, tablets etc.
- **Close** all chat programs, mail clients etc.
- **Keep these things off/away** during the exam.
- If you're found cheating (e.g. giving or receiving help) during or after the exam, you have failed the course.

During and after the exam.

- You are forbidden from sharing any part of your exam solution until the day after the exam.
- You are allowed to **search the Java API documentation here (https://docs.oracle.com/en/java/javase/19/docs/api)** .
  - Otherwise, you may not use any other sources (books, notes, sample codes,

the Internet etc.).
- You are only allowed to use a "simple" text editor (that doesn't have advanced features like code completion or automatic compilation), so no IDEs.
- About the code.
  - Whenever a name is specifically given, use that name exactly.
  - Follow good practices.

Submitting.

- Solve the exercises in order.
- When the time is nearly up (with about 10 minutes left to go), zip the project that you created and upload it into Canvas.

Test cases

Click here to download the required `.jar` file. (Its name has been shortened.)

Compile and run the test cases like this:

```
javac -cp ".;junit5all.jar" exercise/test/ExerciseTestSuite.java
java -jar junit5all.jar -cp . -c exercise.test.ExerciseTestSuite
```

On Linux boxes, use `:` instead of `;`.

If the terminal doesn' support colours and the output is a garbled mess, add the `--disable-ansi-colors` option to the second command.

Download the test cases for the exercise here.

You may not modify the code inside the tester (with the exception below), and by the time you are done with a class, the relevant test case has to be fully "green".

- While your code is incomplete, you may temporarily comment out parts of the tester that reference unfinished parts.
  - This includes the references to the test cases in `ExerciseTestSuite.java`, and also methods in the test cases themselves.

Exercise

Create a program that categorises the letters of a text in the following way.

Character categories (`exercise.counter.Category`)

Create the enum `exercise.counter.Category` with the elements `LETTER`, `WHITESPACE`, and `OTHER`.

Let the class have a static method `determineCategory` that takes a character and returns its `Category`.

- Implement the method in a straightforward manner.
- Hint: use the method **`Character.isLetter(...)`** (https://docs.oracle.com/en/java/javase/19/docs/api/java.base/java/lang/Character.html#isLetter(char)) and **`Character.isWhitespace(...)`**

(https://docs.oracle.com/en/java/javase/19/docs/api/java.base/java/lang/Character.html#isWhitespace(char
.

Simple character counter (`exercise.counter.Counter`)

Create the class `exercise.counter.Counter`.

Let the class have a single field `categories` that is an `ArrayList` that contains `Category` elements. Create a standard getter for it.

- Important: implement all operations so that after construction, you do not change the content of list.

The class has a single constructor with a single `String` field `input`.

- Use `Category.determineCategory` to fill the list with the letters' categories, keeping their order.
- Do *not* store the input text itself.

Create method `getCount` that takes a `Category` object, and returns how many letters in the text belong to that category.

Create method `format` that is visible to the descendants of the class, but nobody else. It takes a `Category` value and returns a text like this: `WHITESPACE: 123`, supposing that 123 is the number of occurrences of whitespace in the text.

Let the standard textual representation of the class look like this: `WHITESPACE: 14↵OTHER: 24↵LETTER: 3` where ↵ is the newline character. Implement the appropriate method by iterating all of the values of `Category`, using the above `format` method. Only put categories with a nonzero frequency into the output.

Pretty printed counter (`exercise.PrettyCounter`)

Create class `exercise.PrettyCounter`, a child of `Counter`. This class produces a visually more appealing representation of the category frequencies. See the sample output at the bottom of the page.

Create two integer fields, `maxWidth` and `maxCount`.

Let the class have a single constructor that takes the text to be processed and the value for `maxWidth`.

- Throw an `IllegalArgumentException` if the text is empty.
- Compute the value for `maxCount` by iterating through the values of `Category`, determining their frequency using the methods of the parent class, and taking the largest value of them all.

Override the method `format` the following way.

Definiáld felül a szülőosztály `format` metódusát az alábbi módon.

- Use `getCount` to produce `count`.
- Then compute `maxWidth * count / maxCount`, making sure that you round up.
- Let the method return this many `#` characters followed by a space and the output

of `format` of the parent class.

Main (`exercise.Main`)

Create `main` in `exercise.Main` that makes a `PrettyCounter` instance and prints it.

Let `PrettyCounter` use the first command line argument as tex and
let `maxWidth` be `30`.

Sample input and output

First argument: `Two households, both alike in dignity, [...]`

- Note: when you pass the parameter, enclose them in quotes (`"`), otherwise the system will split it to many arguments.

Output (the order of the categories could be different from this):

```
############################## LETTER: 31
###### WHITESPACE: 6
####### OTHER: 7
```

Solution.7z (https://canvas.elte.hu/files/2003568/download)

Quiz Score: **0** out of 20