CSE 311: Foundations of Computing I

EXTRA CREDIT: `OnMySourceCodeGENERATOR`

Notation

We write `CODE`(**P**) to mean “the source code of the program **P**”. We also write **P**$(x, y, z, \dots)$ to mean the “the output of the program **P**when given $x, y, z, \dots$ as input”; Note that by output, we mean all the `System.out.print` and `System.out.println` statements (not the return value).

Quines and Identities

- A quine is a program **Q** such that **Q**() = `CODE`(**Q**). That is, it is a program that outputs its own source code when given no input.
- An identity program is a program **Id** such that **Id**$(x) = x$. That is, it is a program that outputs exactly its input.

P`OnMySourceCode`

Let **P** be some particular static method that takes some string as an argument and outputs a string. **P**`OnMySourceCode` is a program that takes no arguments such that

$$\mathbf{P}\texttt{OnMySourceCode}() = \mathbf{P}(\texttt{CODE}(\mathbf{P}\texttt{OnMySourceCode})).$$

Here are some examples of **P**`OnMySourceCode` for various programs **P**:

- If **P**=**Length**(x), then **Length**`OnMySourceCode`() should print the length of its source code.
- If **P**=**AllUpperCase**(x), then **AllUpperCase**`OnMySourceCode`() should print its source code in all upper case letters.

Note that if **P**=**Id**, then **Id**`OnMySourceCode` is just a quine!

`OnMySourceCodeGENERATOR`

Your task for this extra credit is to write the program `OnMySourceCodeGENERATOR`.

`OnMySourceCodeGENERATOR` is a program that given the source code to some particular program **P**, outputs the source code to the program **P**`OnMySourceCode`.

You should name the class that you output to **P**`OnMySourceCode`, and since it should take no input, *main* should execute **P** on **P**`OnMySourceCode`’s source code.

In other words, you will be writing a program that constructs and prints **P**`OnMySourceCode` when given `CODE`(P) as input. Intuitively, we are asking you to write this program instead of directly writing **P**`OnMySourceCode`, because it’s easier to let the computer generate **P**`OnMySourceCode` rather than doing it directly.

Writing Your Program

You are given several Java classes which we explain here:

`OnMySourceCodeGENERATOR.java` This is the file you will be editing. You should not edit main, and you should not edit any other files. This file should write out a class, **P**`OnMySourceCode`, where **P** is the code input to `OnMySourceCodeGENERATOR`using `System.out.println`.

`Tester.java` This program will test your implementation of of `OnMySourceCodeGENERATOR`using `P.input` as the input to your `OnMySourceCodeGENERATOR`function.

`P.input` This file will contain a single (valid) `static` Java method called **P** which takes exactly one `String` argument.

You are permitted, and even encouraged, to add helper functions to the source files to handle repetitive tasks. Output everything using `System.out.print` and `System.out.println`.

> You may not do any disk or network I/O of any kind on this assignment, nor may you use things like the `system` function.

You should only need to use methods from the following Java classes:

- `System.out`
- `String`

Tips!

Long Java string constants: If you have a very long string constant in Java, you can make it out of smaller strings concatenated with a plus sign. Note that in Java you can concatenate ints and chars with string very easily:

```
str = "This is a very long Java string "+
      "constant.  It spans multiple "+
      "lines in the program, but when "+
      "the program is run, this "+
      "will just be one long string "+
      "with "+ 1 + " newline at the end.\n";
```

String Escape Sequences: One way of handling this assignment involves lots of escape sequences (like `\n \t \\ \"`), but there are better ways to do it. In particular, the fact that characters are numbers can be used to deal with this. There is a really cool reading which is both interesting and helpful for this problem; the part most relevant to this assignment is the part about the quine (see Figure 1).