Student: Eduardo de Oliveira Castro
Blazer ID: edc
Date: 03/11/2015

Report for Lab #10

Problem)

Following the given file the student should implement the required methods for getting the input from the user(following the pattern “nameOfTheClass parameter1 parameter2 … parameterX”) and based on these informations create an instance of this object with the proper parameters and using its constructor.

Solution)

The first thing I did was to replace the splitsInput variable(an array) for an instance of StringTokenizer, since it splits the string automatically and has some cool methods for dealing with it.

As the first argument provided by the user is the class I basically took it and used the method forName with this argument and assigned the return in a variable from the type Class. Now we have the class but didn’t initialised it, the next step is deal with the parameters and the constructors. That’s the hard part.

For the parameters I created a new method, createNewInstance, that receives the class and the StringTokenizer instance. For discovering the “anatomy” of the constructor I called the method getConstructors() from the Class object, it returns an array with all the constructors and I assigned it in a Constructor[] variable. Using the for loop I verified all the available constructors to discover one that requires the same number of parameters as provided by the user. If I found one so I create a new array with the types of each parameter from this constructor and a new array of Object[] that will receive values relating to the real type of this parameter(got using the provided getObject() method). After all the method just return a new instance with the list of parameters.

For making sure that the instance is working properly at the last line I print the class of the instance for checking if it is the same as the provided one. If any parameter is provided wrongly then an exception will appear.

Possible Improvements)

Explore better the assertions and exceptions just to make sure that the right types and classes are going in and out. It would also be great to split some complex parts of the code and methods that are performing and taking care of more than one responsibility.