

```
1  /*
2  Noor Haj Dawood - 314997602
3  Bader Daka - 208219212
4  Demo Video - https://youtu.be/ARpuI-boCL8
5  */
6  package il.ac.shenkar.assignments;
7
8  import java.lang.reflect.Field;
9  import java.lang.reflect.Method;
10 import java.util.LinkedList;
11 import java.util.List;
12
13 /**
14  * Displays information about a class: Methods /
15  * Fields / ParentClass
16  * Uses the Class<T> class to get the required
17  * information
18  */
19
20 public class NoorHajDawoodReflectionUtilities
21     implements IReflectionUtilities{
22
23     /**
24      * return a list which includes all the methods
25      * names in className
26      * @param className
27      * @return List<String>
28      * @throws ClassNotFoundException
29      */
30     @Override
31     public List<String> getNamesOfAllMethods(String
32     className) throws ClassNotFoundException {
33         // get Class reference
34         Class<?> clazz = Class.forName(className);
35
36         // Get class Methods, iterate on the results
37         // and add them to the list
38         List<String> list = new LinkedList<String>();
39         Method[] methods = clazz.getDeclaredMethods
40         ();
41         for(Method method: methods) {
42             list.add(method.getName());
43         }
44     }
45 }
```

```

35         }
36         return list;
37     }
38
39     /**
40      * return a list which includes all the fields
names in className
41      * @param className
42      * @return List<String>
43      * @throws ClassNotFoundException
44      */
45     @Override
46     public List<String> getNamesOfAllFields(String
className) throws ClassNotFoundException{
47         // get Class reference
48         Class<?> clazz = Class.forName(className);
49
50         // Get class Methods, iterate on the results
and add them to the list
51         List<String> list = new LinkedList<String>();
52         Field[] fields = clazz.getDeclaredFields();
53         for(Field filed: fields) {
54             list.add(filed.getName());
55         }
56         return list;
57     }
58
59     /**
60      * return the name of the parent class of
className
61      * @param className
62      * @return String
63      * @throws ClassNotFoundException
64      */
65     @Override
66     public String getNameOfParentClass(String
className) throws ClassNotFoundException {
67         // get Class reference
68         Class<?> clazz = Class.forName(className);
69
70         // get Class and then get the Super class an

```

```
70 return it's name
71     return clazz.getClass().getSuperclass().
    getName();
72     }
73 }
74
```