

Intro to Java Week 5 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

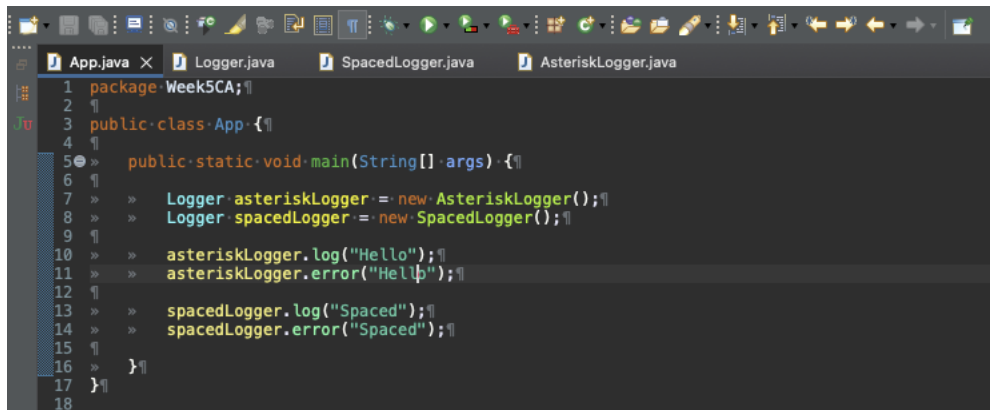
Coding Steps:

1. Create an interface named `Logger`.
2. Add two void methods to the `Logger` interface, each should take a `String` as an argument
 - a. `Log`
 - b. `Error`
3. Create two classes that implement the `Logger` interface
 - a. `AsteriskLogger`
 - b. `SpacedLogger`
4. The `log` method on the `AsteriskLogger` should print out the `String` it receives between 3 asterisks on either side of the `String` (e.g. if the `String` passed in is "Hello", then it should print `***Hello***` to the console.
5. The `error` method on the `AsteriskLogger` should print the `String` it receives inside a box of asterisks, with the `String` preceded by the word "ERROR:". For example, if "Hello" is the argument, the following should be printed:

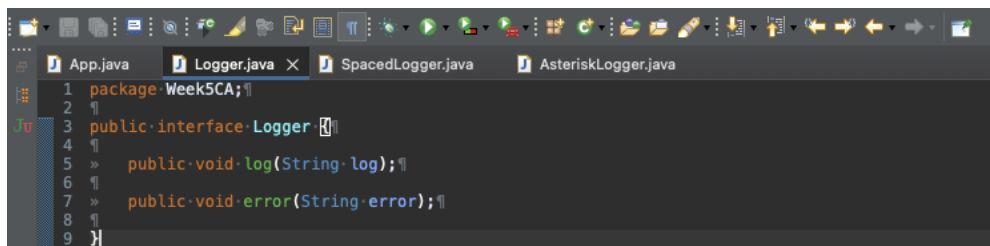
Error: Hello

6. The SpacedLogger should add spaces between each character of the String argument passed into its methods.
7. If the log method received “Hello” as an argument, it should print H e l l o
8. The error method should do the same, but with “ERROR:” preceding the spaced out input (i.e. ERROR: H e l l o)
9. Create a class named App that has a main method.
10. In this class instantiate an instance of each of your logger classes that implement the Logger interface.
11. Test both methods on both instances, passing in Strings of your choice.

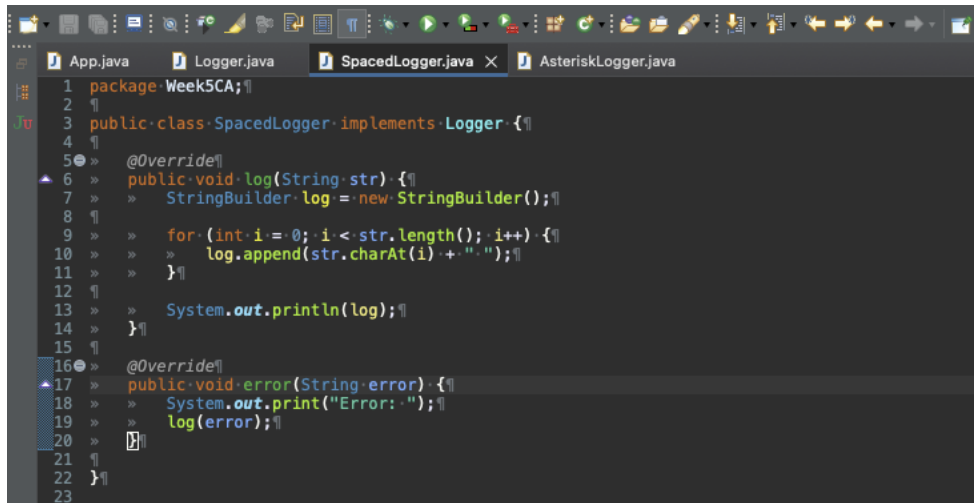
Screenshots of Code:



```
1 package Week5CA;
2
3 public class App {
4
5     public static void main(String[] args) {
6
7         Logger asteriskLogger = new AsteriskLogger();
8         Logger spacedLogger = new SpacedLogger();
9
10        asteriskLogger.log("Hello");
11        asteriskLogger.error("Hello");
12
13        spacedLogger.log("Spaced");
14        spacedLogger.error("Spaced");
15
16    }
17 }
18
```

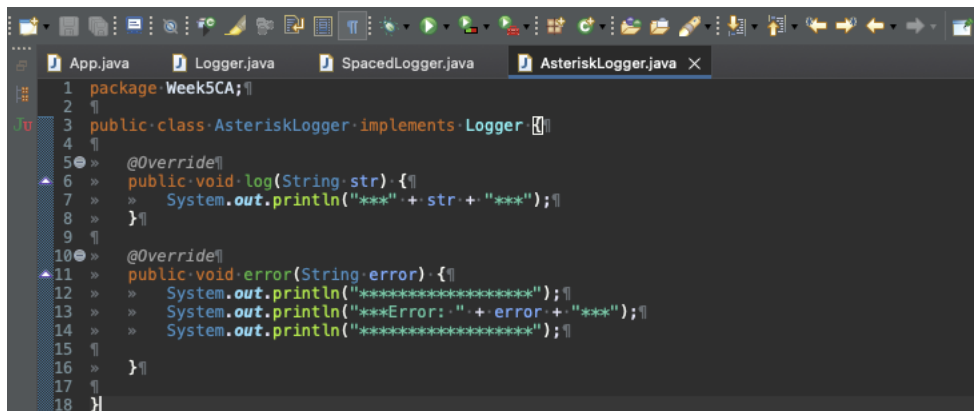


```
1 package Week5CA;
2
3 public interface Logger {
4
5     public void log(String log);
6
7     public void error(String error);
8
9 }
```



This screenshot shows an IDE window with the file `SpacedLogger.java` selected. The code defines a `SpacedLogger` class that implements the `Logger` interface. It includes two methods: `log` and `error`. The `log` method uses a `StringBuilder` to construct a log message with spaces between characters. The `error` method prints the error message with a prefix and then calls the `log` method.

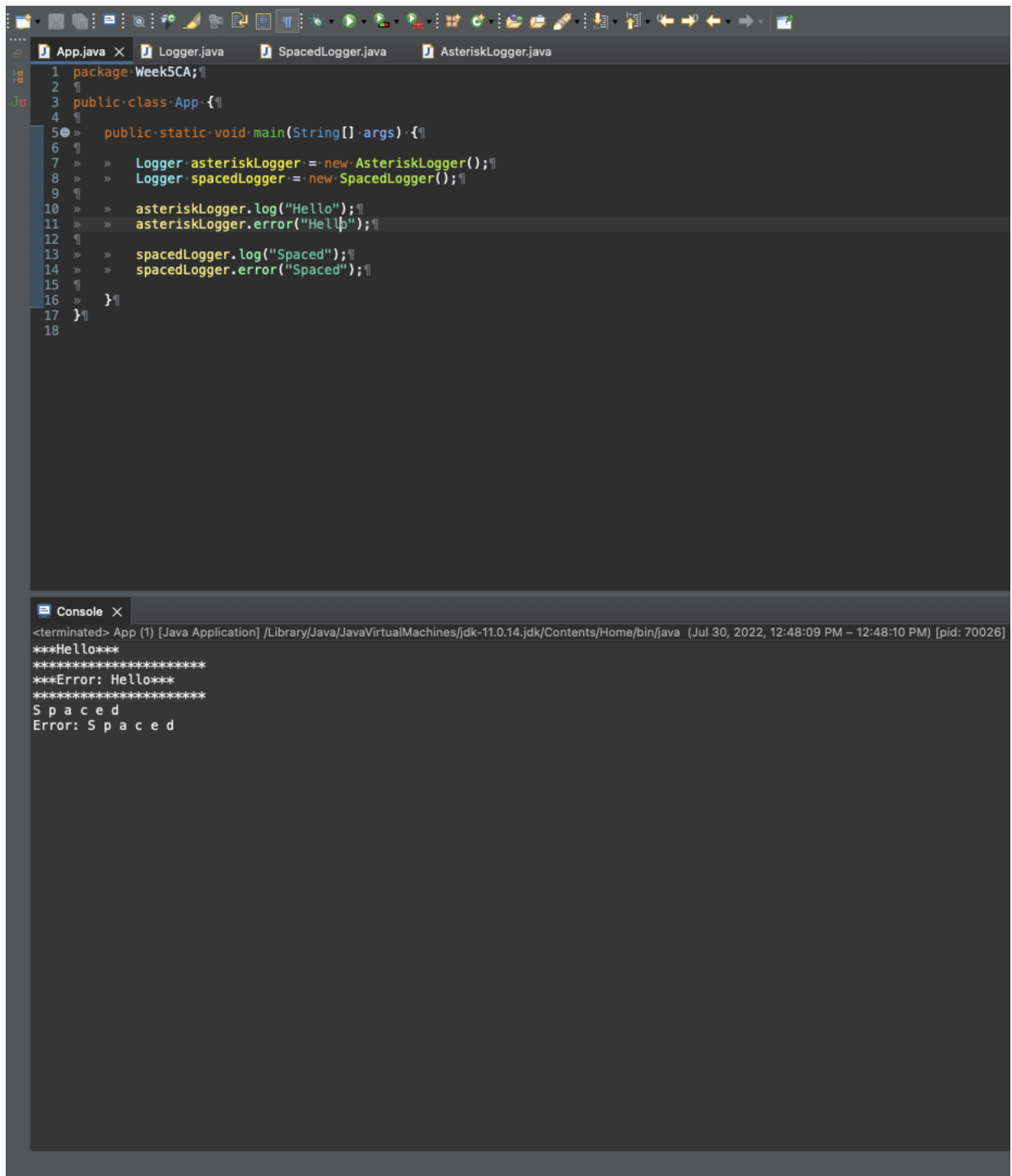
```
1 package Week5CA;
2
3 public class SpacedLogger implements Logger {
4
5     @Override
6     public void log(String str) {
7         StringBuilder log = new StringBuilder();
8
9         for (int i = 0; i < str.length(); i++) {
10             log.append(str.charAt(i) + " ");
11         }
12
13         System.out.println(log);
14     }
15
16     @Override
17     public void error(String error) {
18         System.out.print("Error: ");
19         log(error);
20     }
21 }
22
23
```



This screenshot shows an IDE window with the file `AsteriskLogger.java` selected. The code defines an `AsteriskLogger` class that implements the `Logger` interface. It includes two methods: `log` and `error`. The `log` method prints the log message with asterisks at the beginning and end. The `error` method prints the error message with a prefix, asterisks, and a separator.

```
1 package Week5CA;
2
3 public class AsteriskLogger implements Logger {
4
5     @Override
6     public void log(String str) {
7         System.out.println("***" + str + "***");
8     }
9
10    @Override
11    public void error(String error) {
12        System.out.println("*****");
13        System.out.println("***Error: " + error + "***");
14        System.out.println("*****");
15    }
16
17 }
18
```

Screenshots of Running Application:



The image shows a screenshot of an IDE with four Java files open: App.java, Logger.java, SpacedLogger.java, and AsteriskLogger.java. The App.java file is the active editor, showing a package declaration and a main method. The main method creates instances of AsteriskLogger and SpacedLogger, and then calls log and error methods on them. The console window at the bottom shows the output of the program, which is formatted with asterisks and spaces to match the log messages.

```
1 package Week5CA;
2
3 public class App {
4
5     public static void main(String[] args) {
6
7         Logger.asteriskLogger = new AsteriskLogger();
7         Logger.spacedLogger = new SpacedLogger();
8
9         asteriskLogger.log("Hello");
10        asteriskLogger.error("Hello");
11
12        spacedLogger.log("Spaced");
13        spacedLogger.error("Spaced");
14
15    }
16 }
17
18
```

Console Output:

```
<terminated> App (1) [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.0.14.jdk/Contents/Home/bin/java (Jul 30, 2022, 12:48:09 PM - 12:48:10 PM) [pid: 70026]
***Hello***
*****
***Error: Hello***
*****
S p a c e d
Error: S p a c e d
```

URL to GitHub Repository: