In most programming languages, the compiler carries a preprocessing step to determine if certain statements will compile. For instance it may check to see if parentheses match.

Write a Java program that simulates the actions of a preprocessor, to detect if certain Java constructs are syntactically correct. **Table 1** shows the types of Java statement formats under consideration, and also example of each statement.

Table 1

Format		**Example**
Statement	data_type = expression	int x = 3 + (10 – 4) * (10 + 4)

Method	<attrib> rt name(<parameter>)
	{
	<statement>
	}
		public void display(int n)
		{
		int arr[ ] = new int[n];
		System.out.println(x[2]);
		}

class	<attrib> class Name
	{
	dt fields;
	<attrib> Name(<parameter>)
	{
	method (<parameter>);
	}
	<attrib> rt method(<parameter>)
	{
	}
	}
		public class MyParser
		{
		public static void main(String [ ] arg
		)
		{
		display (10);
		}
		static void display(int x)
		{
		/*
		My pre-processor
		*/
		}

}

Table 2 shows the delimiters under consideration.

Table 2

Delimiters	**Symbol**
Left parenthesis	(
Right parenthesis	)
Left curly braces	{
Right curly braces	}
Left square brackets	[
Right square brackets	]
Forward slash	/
Star (multiplication symbol)	*

Note: In your implementation, design a class called Preprocessor that accepts a file that represents the program to be analyzed. The class will contain, among other possible methods, a method that determines whether or not the statement in the file are valid, with respect to the delimiters of **Table 2**. Do not be concerned with other symbols.

1. You will need a test class. You may want to name it MyPreprocessor.
2. You may have to enter all statements on a single line, unless you will be reading the input from a file, in which case the file would be read using presumable the class BufferedReader or LineNumberReader.
3. Your output would echo the input, and say whether or not the input passed the preprocessing stage.
4. You are to use the concept of **stack** to determine if the constructs are syntactically correct.