Stephanie Harders

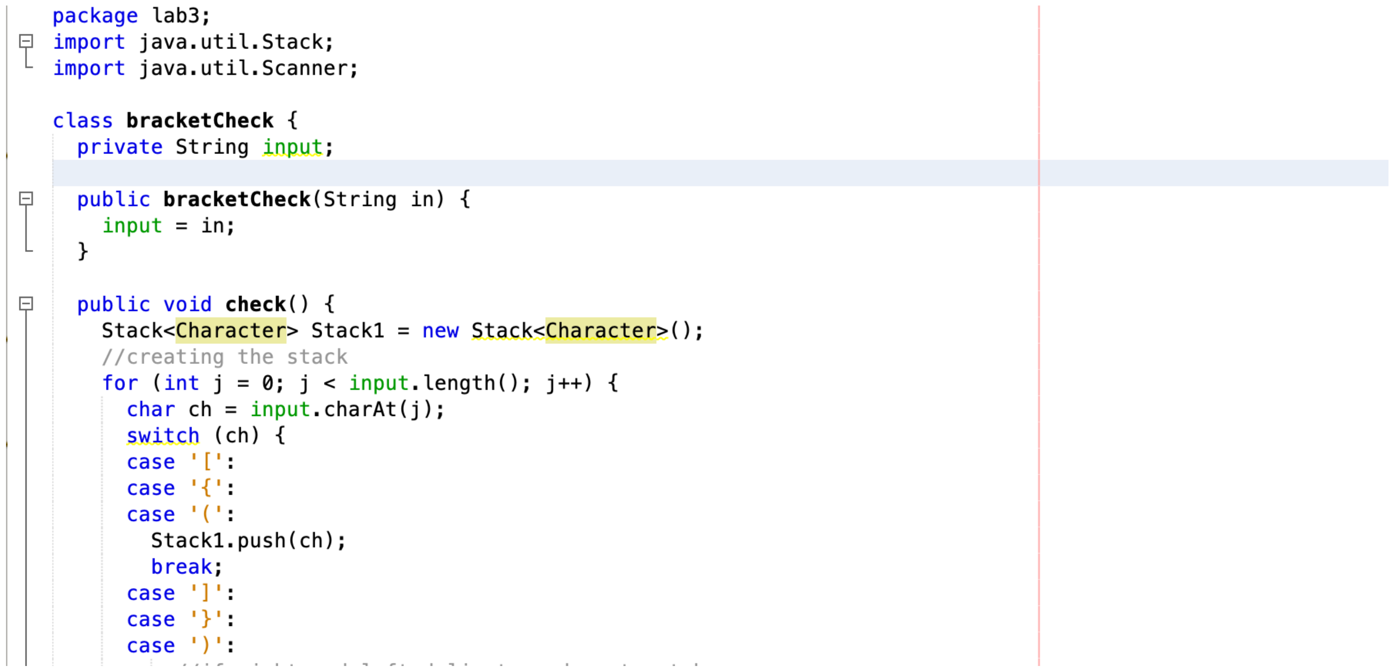
Professor Lane

COM 210

21 April 2023

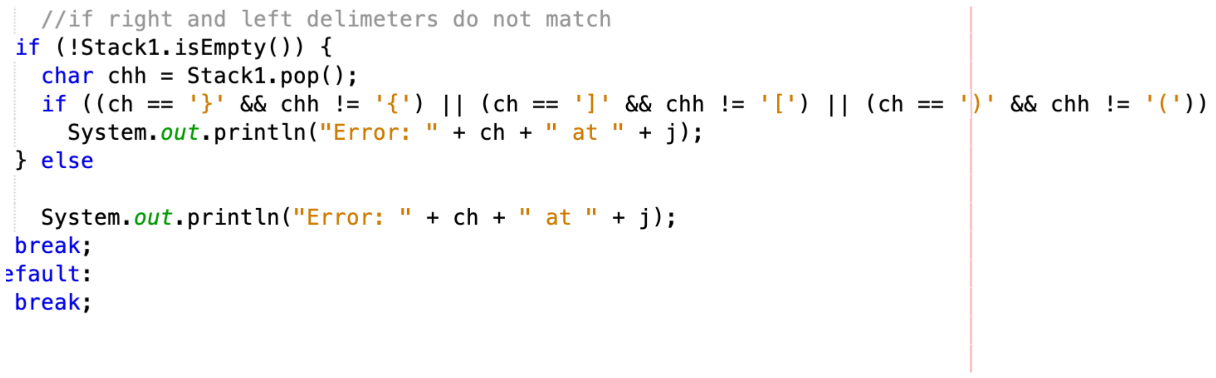
Lab 3

For Lab 3, my partner and I had to create a program that checks the delimiters in a line of text written by a user. This was completed by having the program read characters from the user-inputted string one at a time and placing left (opening) delimiters on a stack. When the program reads a right (closing) delimiter from the user-inputted string, it pops it from the top of the stack and tries to match it with the left delimiter. The following screenshots of the code will further describe our program.



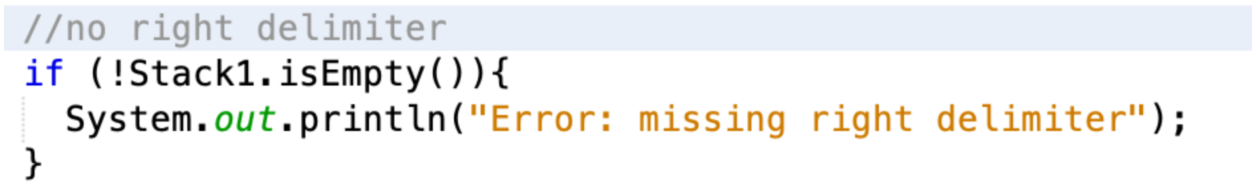
```java
package lab3;
import java.util.Stack;
import java.util.Scanner;

class bracketCheck {
  private String input;

  public bracketCheck(String in) {
    input = in;
  }

  public void check() {
    Stack<Character> Stack1 = new Stack<Character>();
    //creating the stack
    for (int j = 0; j < input.length(); j++) {
      char ch = input.charAt(j);
      switch (ch) {
      case '[':
      case '{':
      case '(':
        Stack1.push(ch);
        break;
      case ']':
      case '}':
      case ')':
```

This section of code showcases how we developed our stack and established cases for our switch statement. We also imported java.util.Stack; so we were able to use methods such as, ‘push’ and ‘pop’.



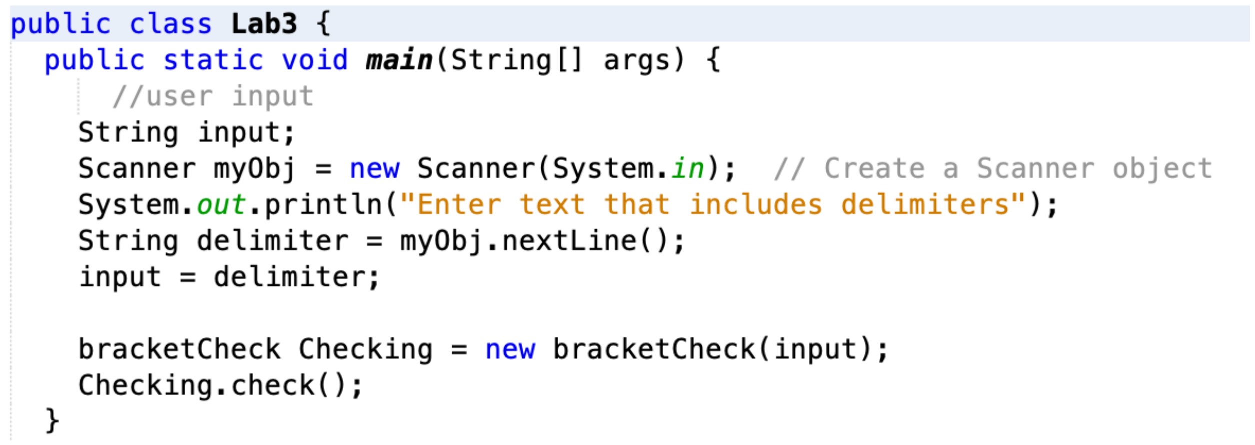
```
  //if right and left delimeters do not match
if (!Stack1.isEmpty()) {
  char chh = Stack1.pop();
  if ((ch == '}' && chh != '{') || (ch == ']' && chh != '[') || (ch == ')' && chh != '('))
    System.out.println("Error: " + ch + " at " + j);
} else

  System.out.println("Error: " + ch + " at " + j);
break;
efault:
break;
```

These lines of code are our if statements. The first one pops the right (closing) delimiter from the user-inputted string to the top of the stack and tries to match it with the left delimiter. If the left delimiter does not match an error statement will print. An example of this would be: “Error:] at 3”.



```
//no right delimiter
if (!Stack1.isEmpty()){
  System.out.println("Error: missing right delimiter");
}
```

This error message will occur if no right delimiter is read from the user-inputted string.



```
public class Lab3 {
  public static void main(String[] args) {
      //user input
    String input;
    Scanner myObj = new Scanner(System.in);  // Create a Scanner object
    System.out.println("Enter text that includes delimiters");
    String delimiter = myObj.nextLine();
    input = delimiter;

    bracketCheck Checking = new bracketCheck(input);
    Checking.check();
  }
```

The final section of the code is how my partner and I received a user-inputted string/text. To eliminate any confusion, we specified that the text must include delimiters. This ensures that the program will serve its' purpose. When a string is inputted, the program reads the characters one at a time and places left (opening) delimiters on a stack. The previously discussed if statements are then applied to see whether or not the left and right delimiters match one another. Or if there is a right (closing) delimiter at all.

Working on this assignment allowed my partner and me to comprehend stacks and their operations much more. Implementing our code and understanding errors created a more thorough learning experience in this subject.