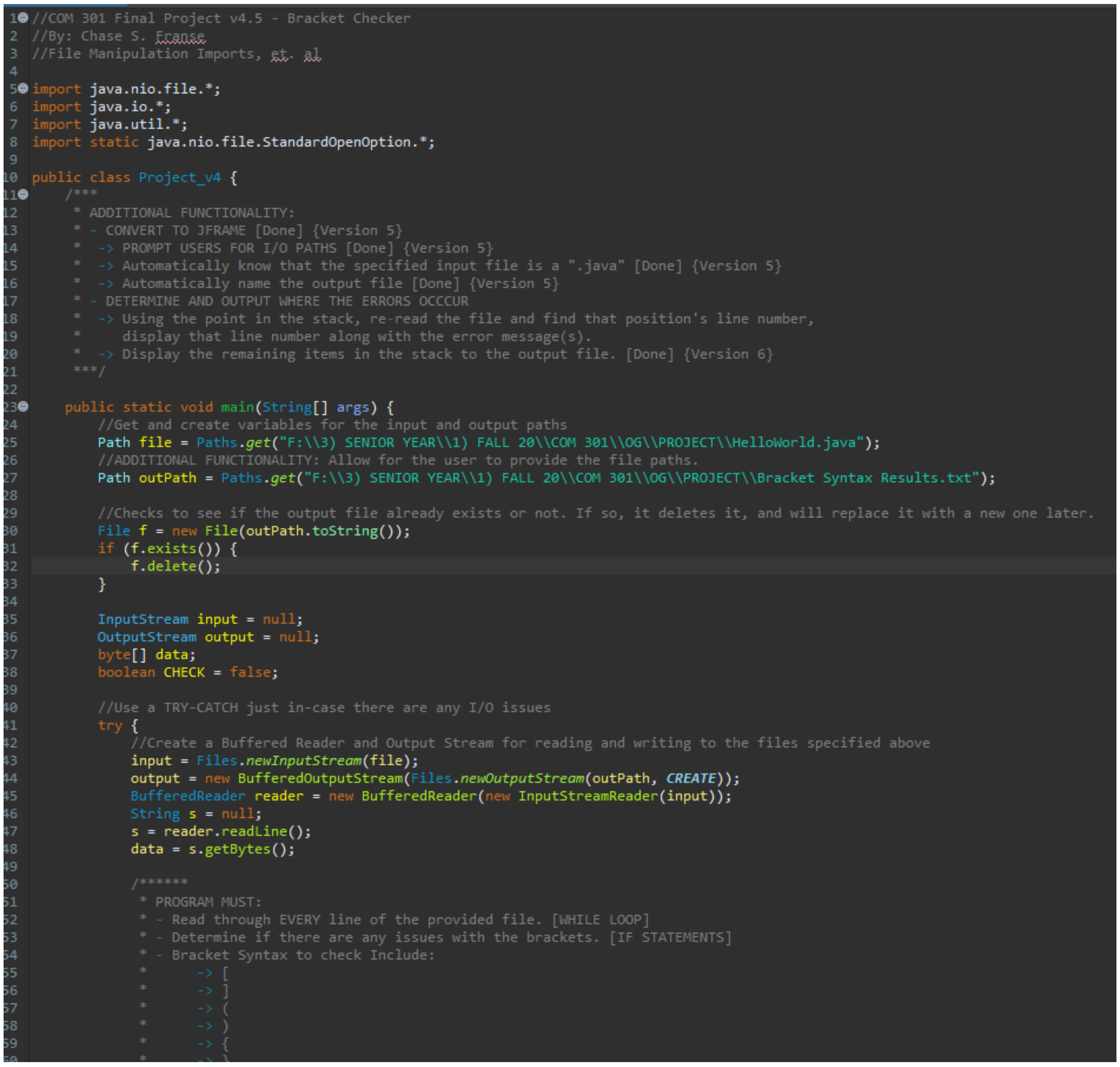
Chase Franse
Dr. Hong
COM 301
20 November 2020

Final Project Screenshots:

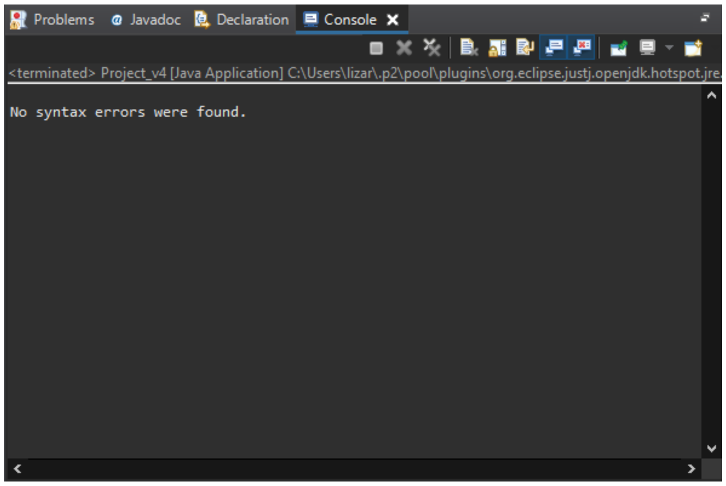
- Project_v4
- Project_v5_EXE
- Project_v6

Project_v4:



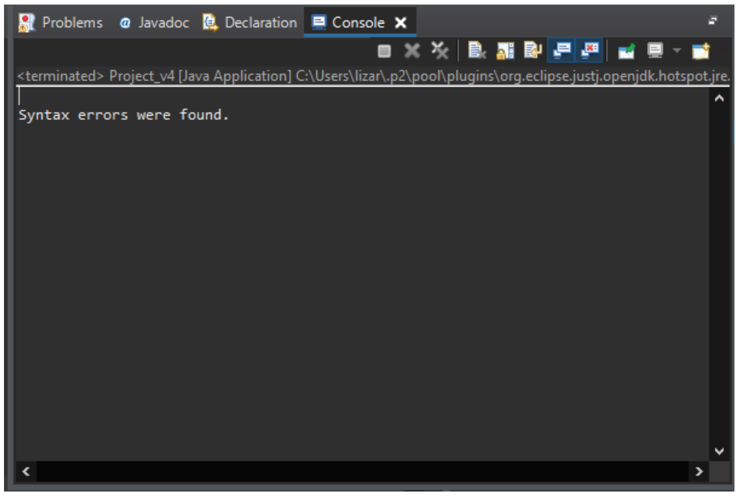
```java
//COM 301 Final Project v4.5 - Bracket Checker
//By: Chase S. Franse
//File Manipulation Imports, et. al

import java.nio.file.*;
import java.io.*;
import java.util.*;
import static java.nio.file.StandardOpenOption.*;

public class Project_v4 {
    /***
     * ADDITIONAL FUNCTIONALITY:
     * - CONVERT TO JFRAME [Done] {Version 5}
     *  -> PROMPT USERS FOR I/O PATHS [Done] {Version 5}
     *  -> Automatically know that the specified input file is a ".java" [Done] {Version 5}
     *  -> Automatically name the output file [Done] {Version 5}
     * - DETERMINE AND OUTPUT WHERE THE ERRORS OCCCUR
     *  -> Using the point in the stack, re-read the file and find that position's line number,
     *     display that line number along with the error message(s).
     *  -> Display the remaining items in the stack to the output file. [Done] {Version 6}
     ***/

    public static void main(String[] args) {
        //Get and create variables for the input and output paths
        Path file = Paths.get("F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\HelloWorld.java");
        //ADDITIONAL FUNCTIONALITY: Allow for the user to provide the file paths.
        Path outPath = Paths.get("F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\Bracket Syntax Results.txt");

        //Checks to see if the output file already exists or not. If so, it deletes it, and will replace it with a new one later.
        File f = new File(outPath.toString());
        if (f.exists()) {
            f.delete();
        }

        InputStream input = null;
        OutputStream output = null;
        byte[] data;
        boolean CHECK = false;

        //Use a TRY-CATCH just in-case there are any I/O issues
        try {
            //Create a Buffered Reader and Output Stream for reading and writing to the files specified above
            input = Files.newInputStream(file);
            output = new BufferedOutputStream(Files.newOutputStream(outPath, CREATE));
            BufferedReader reader = new BufferedReader(new InputStreamReader(input));
            String s = null;
            s = reader.readLine();
            data = s.getBytes();

            /******
             * PROGRAM MUST:
             * - Read through EVERY line of the provided file. [WHILE LOOP]
             * - Determine if there are any issues with the brackets. [IF STATEMENTS]
             * - Bracket Syntax to check Include:
             *      -> [
             *      -> ]
             *      -> (
             *      -> )
             *      -> {
             *      -> }
```

HelloWorld.java:



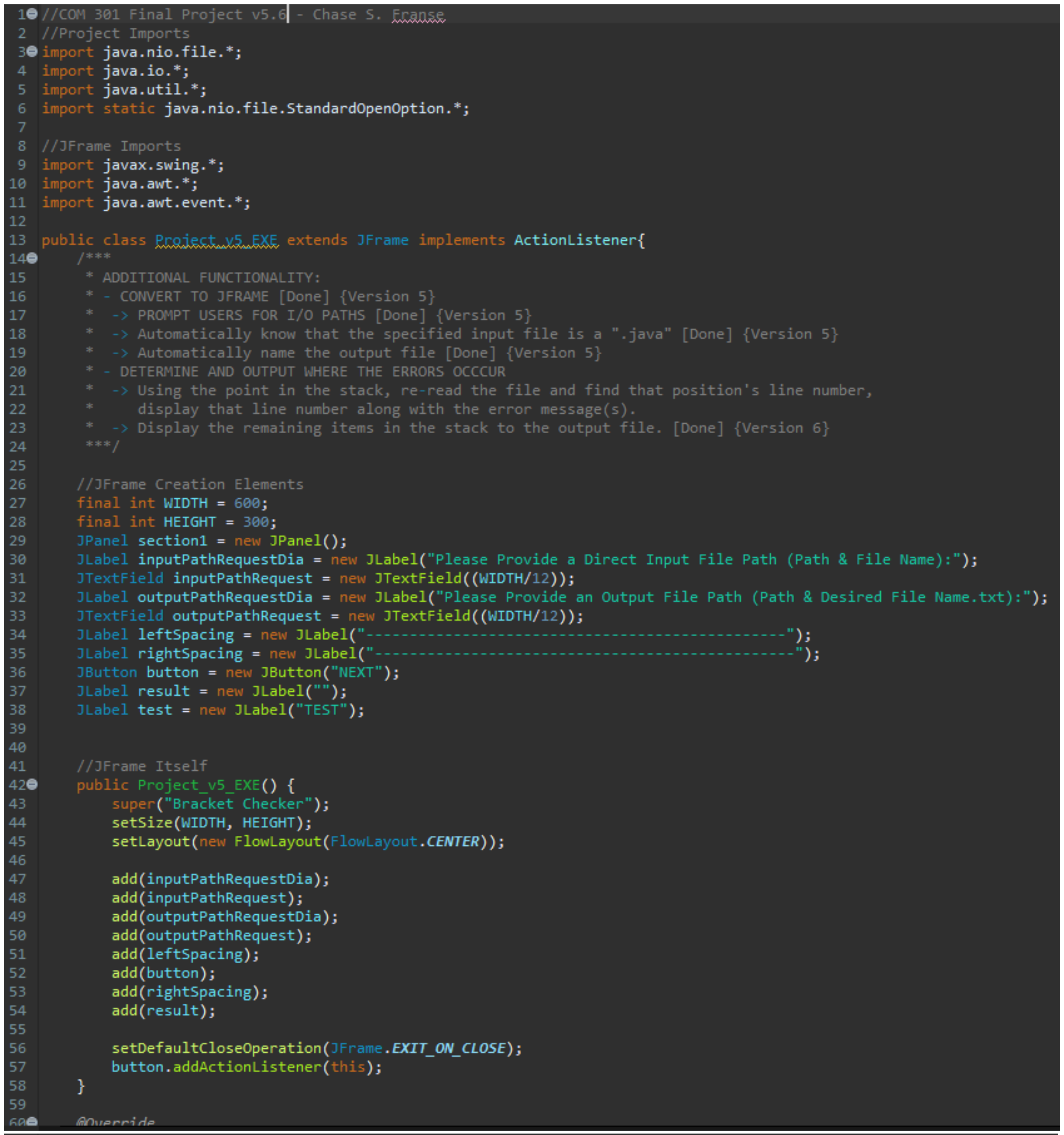
```
<terminated> Project_v4 [Java Application] C:\Users\lizar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.
No syntax errors were found.
```

HelloWorld_wrong.java:

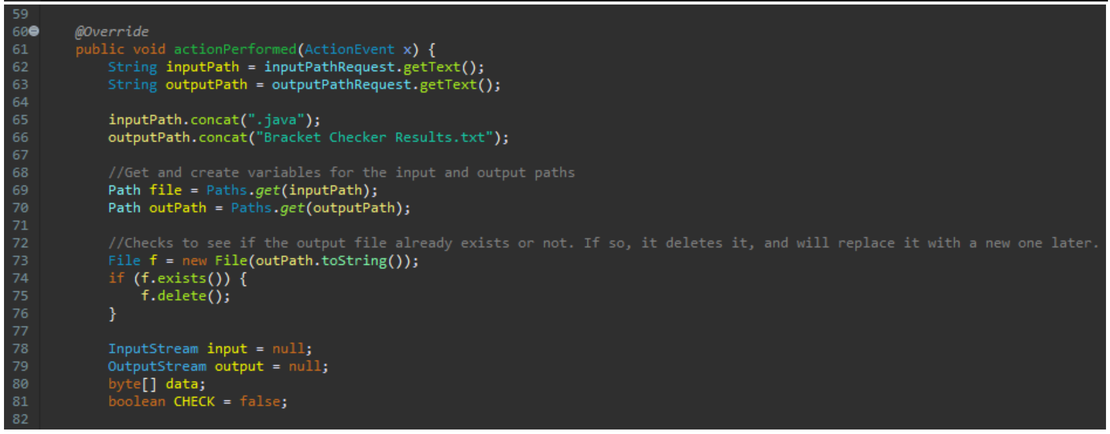


```
<terminated> Project_v4 [Java Application] C:\Users\lizar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.
Syntax errors were found.
```

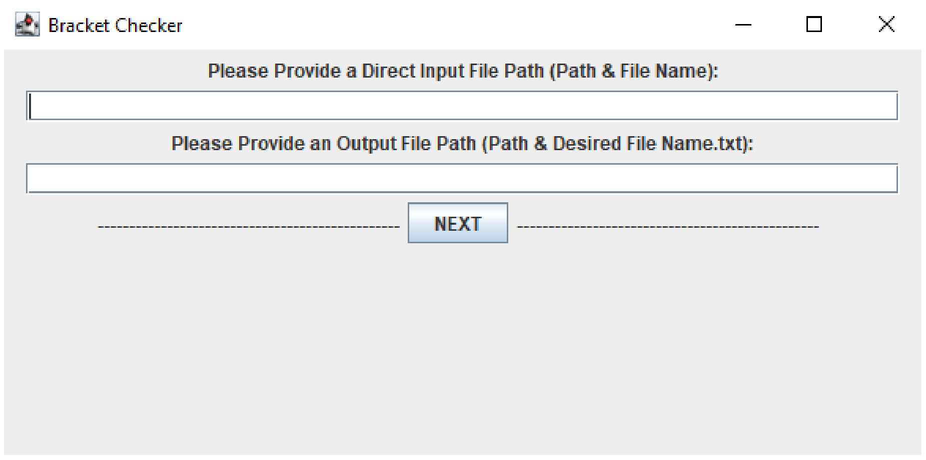
Project_v5:



```java
//COM 301 Final Project v5.6 - Chase S. Franse
//Project Imports
import java.nio.file.*;
import java.io.*;
import java.util.*;
import static java.nio.file.StandardOpenOption.*;

//JFrame Imports
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Project_v5_EXE extends JFrame implements ActionListener{
    /***
     * ADDITIONAL FUNCTIONALITY:
     * - CONVERT TO JFRAME [Done] {Version 5}
     *  -> PROMPT USERS FOR I/O PATHS [Done] {Version 5}
     *  -> Automatically know that the specified input file is a ".java" [Done] {Version 5}
     *  -> Automatically name the output file [Done] {Version 5}
     * - DETERMINE AND OUTPUT WHERE THE ERRORS OCCCUR
     *  -> Using the point in the stack, re-read the file and find that position's line number,
     *     display that line number along with the error message(s).
     *  -> Display the remaining items in the stack to the output file. [Done] {Version 6}
     ***/

    //JFrame Creation Elements
    final int WIDTH = 600;
    final int HEIGHT = 300;
    JPanel section1 = new JPanel();
    JLabel inputPathRequestDia = new JLabel("Please Provide a Direct Input File Path (Path & File Name):");
    JTextField inputPathRequest = new JTextField((WIDTH/12));
    JLabel outputPathRequestDia = new JLabel("Please Provide an Output File Path (Path & Desired File Name.txt):");
    JTextField outputPathRequest = new JTextField((WIDTH/12));
    JLabel leftSpacing = new JLabel("-------------------------------------------------");
    JLabel rightSpacing = new JLabel("-------------------------------------------------");
    JButton button = new JButton("NEXT");
    JLabel result = new JLabel("");
    JLabel test = new JLabel("TEST");


    //JFrame Itself
    public Project_v5_EXE() {
        super("Bracket Checker");
        setSize(WIDTH, HEIGHT);
        setLayout(new FlowLayout(FlowLayout.CENTER));

        add(inputPathRequestDia);
        add(inputPathRequest);
        add(outputPathRequestDia);
        add(outputPathRequest);
        add(leftSpacing);
        add(button);
        add(rightSpacing);
        add(result);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        button.addActionListener(this);
    }

```



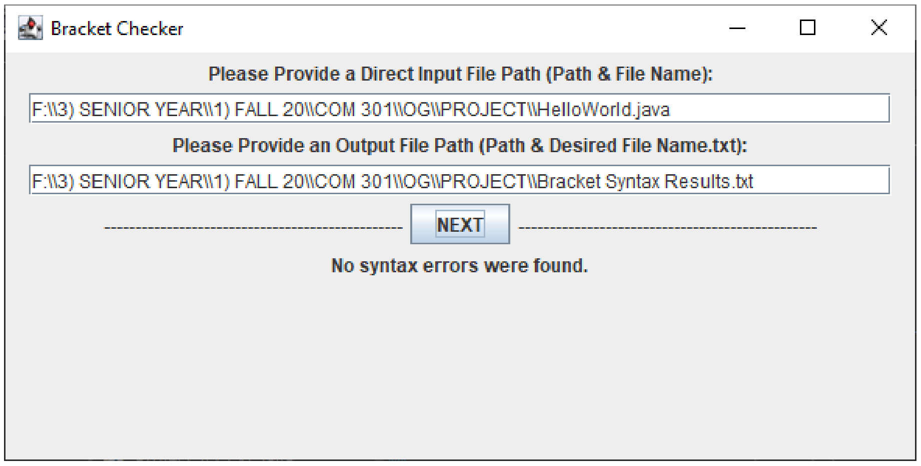
```java

    @Override
    public void actionPerformed(ActionEvent x) {
        String inputPath = inputPathRequest.getText();
        String outputPath = outputPathRequest.getText();

        inputPath.concat(".java");
        outputPath.concat("Bracket Checker Results.txt");

        //Get and create variables for the input and output paths
        Path file = Paths.get(inputPath);
        Path outPath = Paths.get(outputPath);

        //Checks to see if the output file already exists or not. If so, it deletes it, and will replace it with a new one later.
        File f = new File(outPath.toString());
        if (f.exists()) {
            f.delete();
        }

        InputStream input = null;
        OutputStream output = null;
        byte[] data;
        boolean CHECK = false;

```



Please Provide a Direct Input File Path (Path & File Name):

Please Provide an Output File Path (Path & Desired File Name.txt):

HelloWorld.java:



Please Provide a Direct Input File Path (Path & File Name):

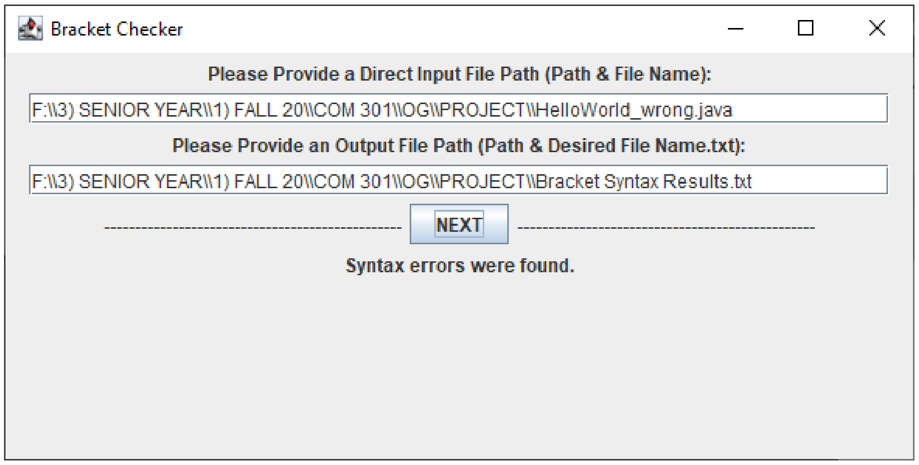
F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\HelloWorld.java

Please Provide an Output File Path (Path & Desired File Name.txt):

F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\Bracket Syntax Results.txt

No syntax errors were found.

HelloWorld_wrong.java:



Please Provide a Direct Input File Path (Path & File Name):

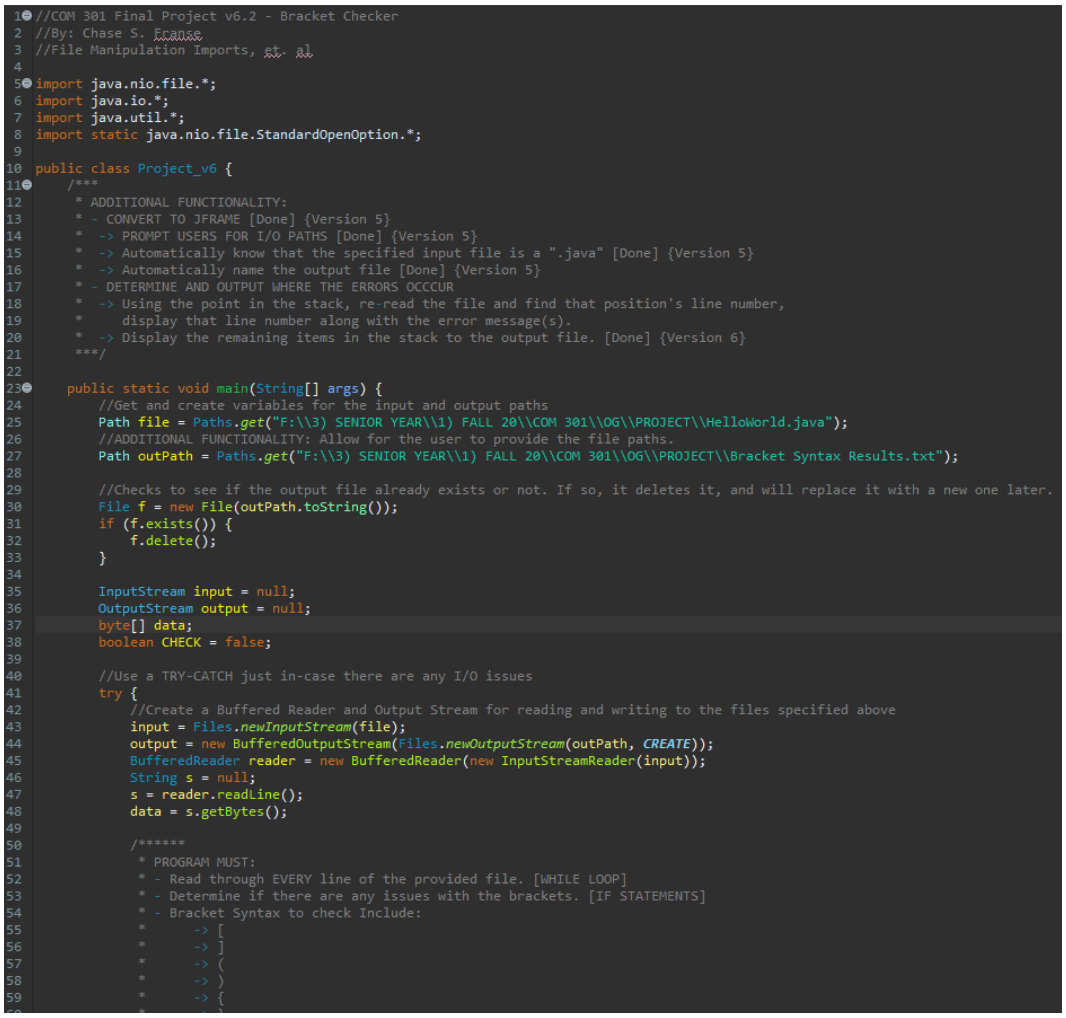
F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\HelloWorld_wrong.java

Please Provide an Output File Path (Path & Desired File Name.txt):

F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\Bracket Syntax Results.txt

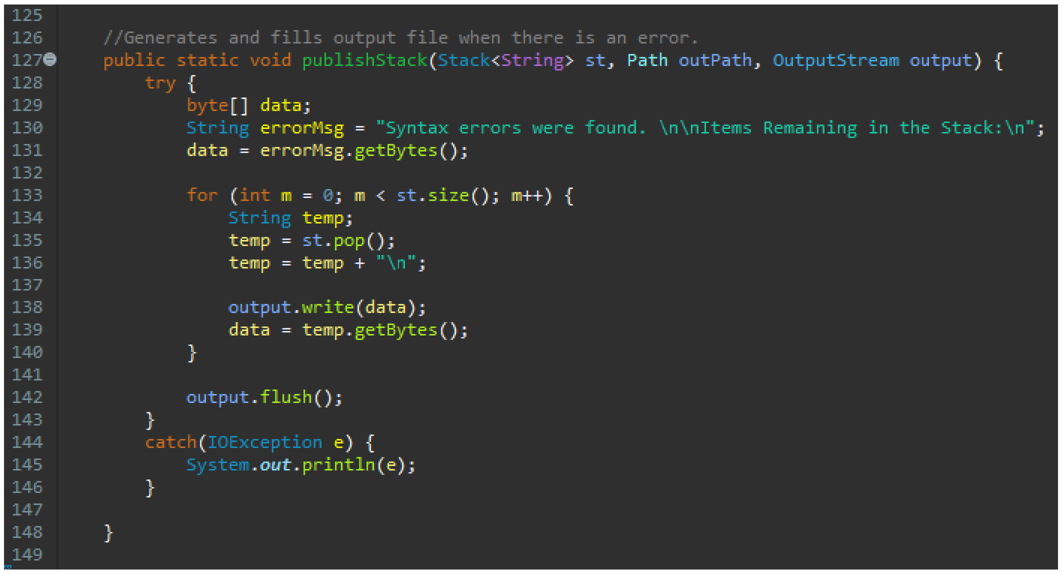
Syntax errors were found.

Project_v6:



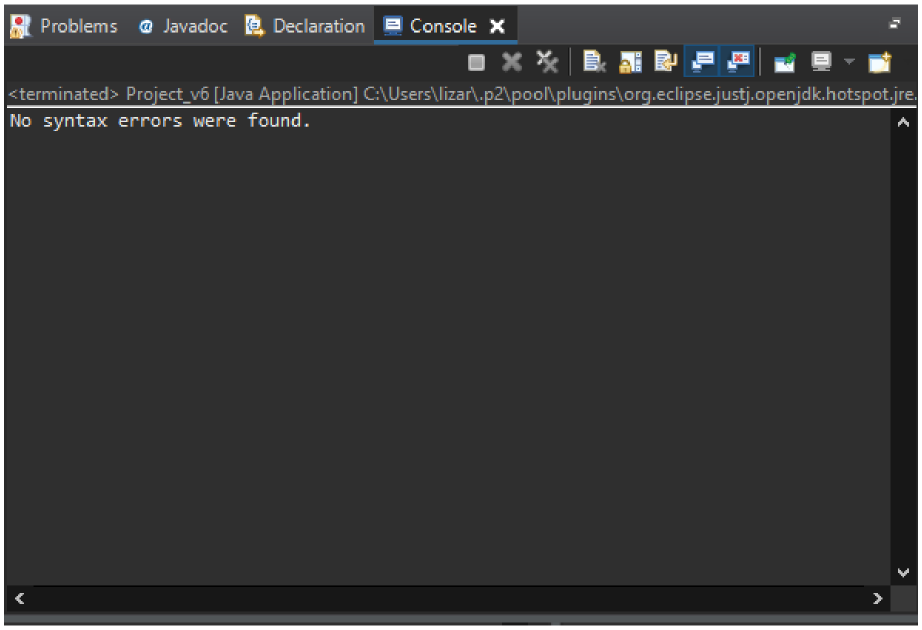
```java
//COM 301 Final Project v6.2 - Bracket Checker
//By: Chase S. Franse
//File Manipulation Imports, et. al

import java.nio.file.*;
import java.io.*;
import java.util.*;
import static java.nio.file.StandardOpenOption.*;

public class Project_v6 {
    /***
     * ADDITIONAL FUNCTIONALITY:
     * - CONVERT TO JFRAME [Done] {Version 5}
     *  -> PROMPT USERS FOR I/O PATHS [Done] {Version 5}
     *  -> Automatically know that the specified input file is a ".java" [Done] {Version 5}
     *  -> Automatically name the output file [Done] {Version 5}
     * - DETERMINE AND OUTPUT WHERE THE ERRORS OCCCUR
     *  -> Using the point in the stack, re-read the file and find that position's line number,
     *     display that line number along with the error message(s).
     *  -> Display the remaining items in the stack to the output file. [Done] {Version 6}
     ***/

    public static void main(String[] args) {
        //Get and create variables for the input and output paths
        Path file = Paths.get("F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\HelloWorld.java");
        //ADDITIONAL FUNCTIONALITY: Allow for the user to provide the file paths.
        Path outPath = Paths.get("F:\\3) SENIOR YEAR\\1) FALL 20\\COM 301\\OG\\PROJECT\\Bracket Syntax Results.txt");

        //Checks to see if the output file already exists or not. If so, it deletes it, and will replace it with a new one later.
        File f = new File(outPath.toString());
        if (f.exists()) {
            f.delete();
        }

        InputStream input = null;
        OutputStream output = null;
        byte[] data;
        boolean CHECK = false;

        //Use a TRY-CATCH just in-case there are any I/O issues
        try {
            //Create a Buffered Reader and Output Stream for reading and writing to the files specified above
            input = Files.newInputStream(file);
            output = new BufferedOutputStream(Files.newOutputStream(outPath, CREATE));
            BufferedReader reader = new BufferedReader(new InputStreamReader(input));
            String s = null;
            s = reader.readLine();
            data = s.getBytes();

            /******
             * PROGRAM MUST:
             * - Read through EVERY line of the provided file. [WHILE LOOP]
             * - Determine if there are any issues with the brackets. [IF STATEMENTS]
             * - Bracket Syntax to check Include:
             *      -> [
             *      -> ]
             *      -> (
             *      -> )
             *      -> {
```

…



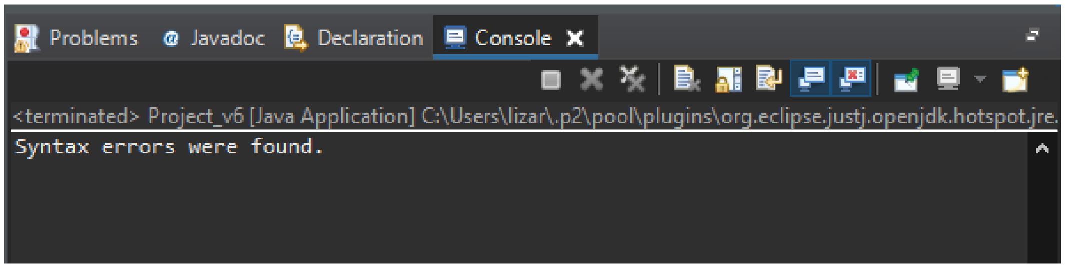
```java

    //Generates and fills output file when there is an error.
    public static void publishStack(Stack<String> st, Path outPath, OutputStream output) {
        try {
            byte[] data;
            String errorMsg = "Syntax errors were found. \n\nItems Remaining in the Stack:\n";
            data = errorMsg.getBytes();

            for (int m = 0; m < st.size(); m++) {
                String temp;
                temp = st.pop();
                temp = temp + "\n";

                output.write(data);
                data = temp.getBytes();
            }

            output.flush();
        }
        catch(IOException e) {
            System.out.println(e);
        }

    }

```

HelloWorld.java:

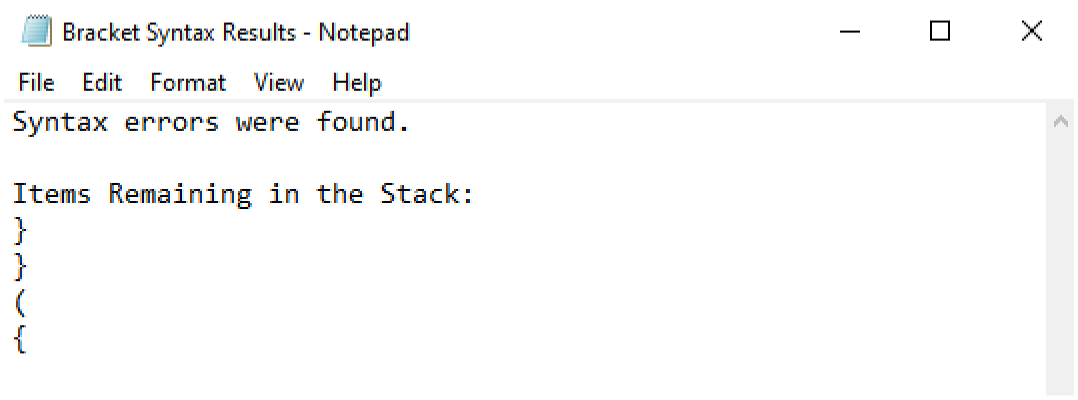


```
<terminated> Project_v6 [Java Application] C:\Users\lizar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre
No syntax errors were found.
```

HelloWorld_wrong.java:



```
<terminated> Project_v6 [Java Application] C:\Users\lizar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre
Syntax errors were found.
```



```
Syntax errors were found.

Items Remaining in the Stack:
}
}
(
{
```