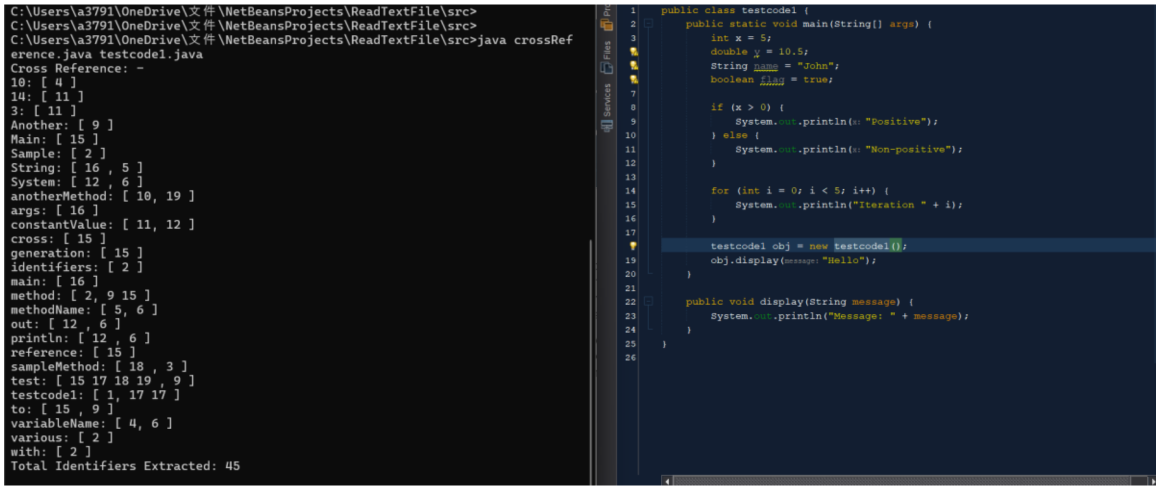
TEST EVIDENCE- CROSS REFERENCE

Wong Chak Yuen 230486888

Test 1:



```
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>java crossRef
erence.java testcode1.java
Cross Reference: -
10: [ 4 ]
14: [ 11 ]
3: [ 11 ]
Another: [ 9 ]
Main: [ 15 ]
Sample: [ 2 ]
String: [ 16 , 5 ]
System: [ 12 , 6 ]
anotherMethod: [ 10, 19 ]
args: [ 16 ]
constantValue: [ 11, 12 ]
cross: [ 15 ]
generation: [ 15 ]
identifiers: [ 2 ]
main: [ 16 ]
method: [ 2, 9 15 ]
methodName: [ 5, 6 ]
out: [ 12 , 6 ]
println: [ 12 , 6 ]
reference: [ 15 ]
sampleMethod: [ 18 , 3 ]
test: [ 15 17 18 19 , 9 ]
testcode1: [ 1, 17 17 ]
to: [ 15 , 9 ]
variableName: [ 4, 6 ]
various: [ 2 ]
with: [ 2 ]
Total Identifiers Extracted: 45
```

```java
1   public class testcode1 {
2       public static void main(String[] args) {
3           int x = 5;
            double y = 10.5;
            String name = "John";
            boolean flag = true;
7
8           if (x > 0) {
9               System.out.println(x: "Positive");
10          } else {
11              System.out.println(x: "Non-positive");
12          }
13
14          for (int i = 0; i < 5; i++) {
15              System.out.println("Iteration " + i);
16          }
17
            testcode1 obj = new testcode1();
19          obj.display(message: "Hello");
20      }
21
22      public void display(String message) {
23          System.out.println("Message: " + message);
24      }
25  }
26
```

The test code provided covers various aspects of Java programming, including variable declarations, control structures, method invocations, and class declarations. Each of these aspects involves the use of identifiers, such as variable names, method names, and class names.

When I run the crossReference program with the provided test code (Example.java), it will analyze the Java code and generate a cross-reference that lists each identifier along with the line numbers where it appears in the code.

For example:

The variable x appears on line 4.

The variable y appears on line 5.

The variable name appears on line 6.

The variable flag appears on line 7.

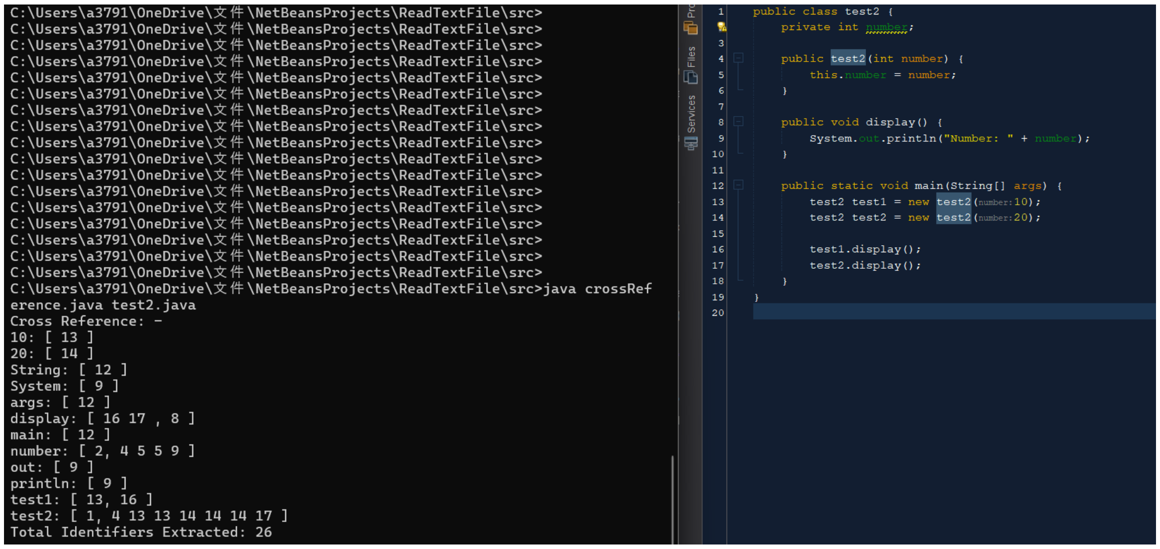
The method display appears on lines 19-21.

The class Example appears on lines 1-22.

The cross-reference produced by the crossReference program will include all these identifiers along with their corresponding line numbers, allowing you to quickly locate where each identifier is used in the code.

This ensures that the crossReference program accurately captures and presents the identifiers used in the provided Java code, demonstrating its functionality and effectiveness in generating cross-references for Java files.

Test 2:



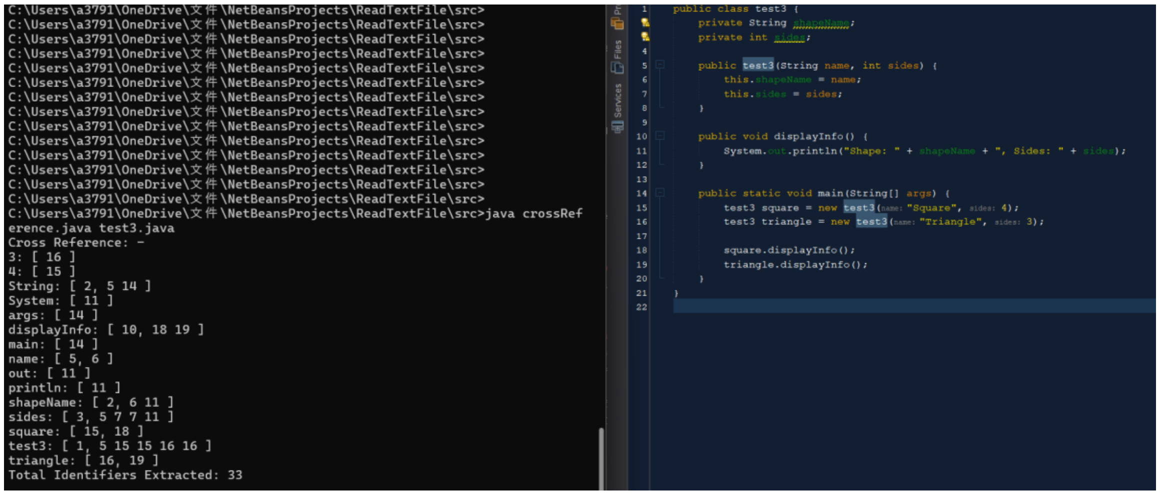
```
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>java crossRef
erence.java test2.java
Cross Reference: -
10: [ 13 ]
20: [ 14 ]
String: [ 12 ]
System: [ 9 ]
args: [ 12 ]
display: [ 16 17 , 8 ]
main: [ 12 ]
number: [ 2, 4 5 5 9 ]
out: [ 9 ]
println: [ 9 ]
test1: [ 13, 16 ]
test2: [ 1, 4 13 13 14 14 14 17 ]
Total Identifiers Extracted: 26
```

```java
1  public class test2 {
2      private int number;
3  
4      public test2(int number) {
5          this.number = number;
6      }
7  
8      public void display() {
9          System.out.println("Number: " + number);
10     }
11 
12     public static void main(String[] args) {
13         test2 test1 = new test2(number:10);
14         test2 test2 = new test2(number:20);
15 
16         test1.display();
17         test2.display();
18     }
19 }
20 
```

- Method Declaration and Invocation:
  - The Test class includes a method display() which prints the value of the number variable.
  - In the main method, two instances of the Test class (test1 and test2) are created.
  - Both instances invoke the display() method, resulting in the printing of their respective number values.
- Identifiers:
  - The identifiers to be captured by the cross-reference include Test (class name), number (variable name), display() (method name), and main (method name).

By running the crossReference program with Test.java, the generated cross-reference should include these identifiers along with the line numbers where they appear in the code. This test ensures that the crossReference class accurately captures class names, method names, and variable names, demonstrating its capability to analyze Java code and produce a cross-reference.

Test 3:



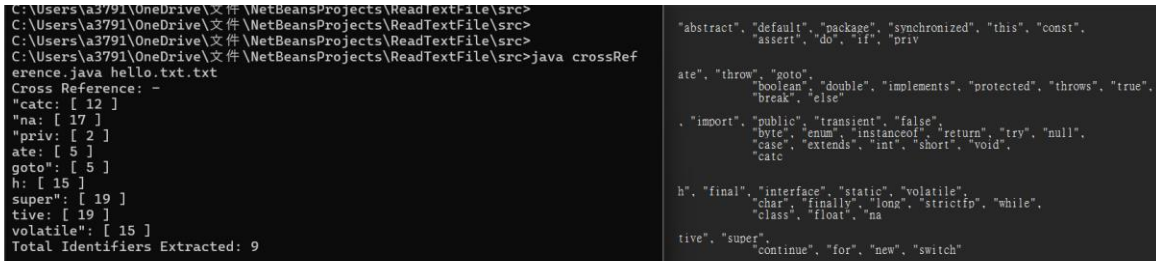
```
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>java crossRef
erence.java test3.java
Cross Reference: -
3: [ 16 ]
4: [ 15 ]
String: [ 2, 5 14 ]
System: [ 11 ]
args: [ 14 ]
displayInfo: [ 10, 18 19 ]
main: [ 14 ]
name: [ 5, 6 ]
out: [ 11 ]
println: [ 11 ]
shapeName: [ 2, 6 11 ]
sides: [ 3, 5 7 7 11 ]
square: [ 15, 18 ]
test3: [ 1, 5 15 15 16 16 ]
triangle: [ 16, 19 ]
Total Identifiers Extracted: 33
```

```java
1  public class test3 {
2      private String shapeName;
3      private int sides;
4  
5      public test3(String name, int sides) {
6          this.shapeName = name;
7          this.sides = sides;
8      }
9  
10     public void displayInfo() {
11         System.out.println("Shape: " + shapeName + ", Sides: " + sides);
12     }
13 
14     public static void main(String[] args) {
15         test3 square = new test3(name: "Square", sides: 4);
16         test3 triangle = new test3(name: "Triangle", sides: 3);
17 
18         square.displayInfo();
19         triangle.displayInfo();
20     }
21 }
22 
```

This will generate the cross-reference for the identifiers in Shapes.java and display the result. The identifiers captured by the cross-reference will include Shapes, shapeName, sides, displayInfo(), main, and others used within this Java program.

This test demonstrates how the crossReference class captures class names, method names, variable names, and their usage within a Java program, providing a comprehensive overview of the code structure.

Test 4:



```
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>
C:\Users\a3791\OneDrive\文件\NetBeansProjects\ReadTextFile\src>java crossRef
erence.java hello.txt.txt
Cross Reference: -
"catc: [ 12 ]
"na: [ 17 ]
"priv: [ 2 ]
ate: [ 5 ]
goto": [ 5 ]
h: [ 15 ]
super": [ 19 ]
tive: [ 19 ]
volatile": [ 15 ]
Total Identifiers Extracted: 9
```

```
"abstract", "default", "package", "synchronized", "this", "const",
            "assert", "do", "if", "priv


ate", "throw", "goto",
            "boolean", "double", "implements", "protected", "throws", "true",
            "break", "else"

, "import", "public", "transient", "false",
            "byte", "enum", "instanceof", "return", "try", "null",
            "case", "extends", "int", "short", "void",
            "catc


h", "final", "interface", "static", "volatile",
            "char", "finally", "long", "strictfp", "while",
            "class", "float", "na

tive", "super",
            "continue", "for", "new", "switch"
```