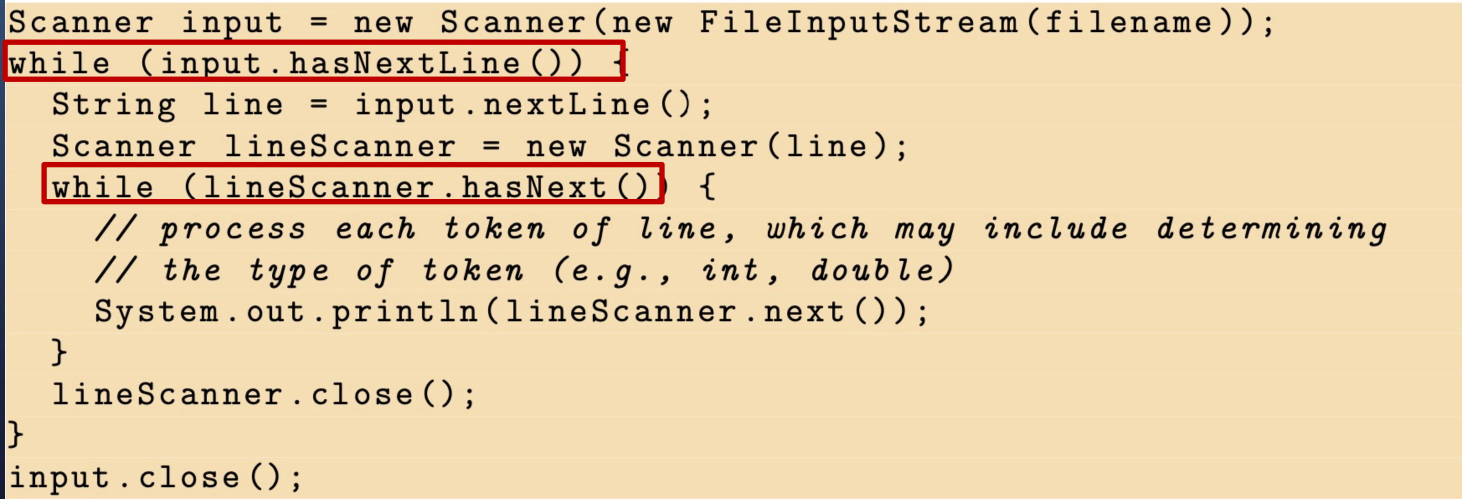
FILE INPUT (LINE-BASED PROCESSING)

zyBook 8.3
Scanner API, FileInputStream API, FileNotFoundException API
Oracle - Java Tutorial: I/O Streams

@ Dr. Jessica Young Schmidt and NCSU Computer Science Faculty

LINE-BASED PROCESSING

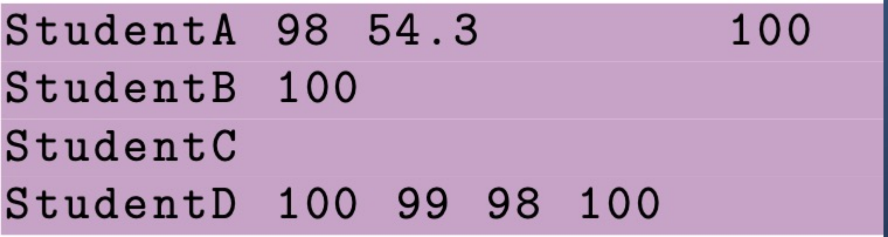
- Alternative to token-based processing
- Process input line by line
  - Use nextLine() and hasNextLine() methods from Scanner class
  - We can pass a String to a Scanner to process through a line of text.
- Keeps white space and line breaks of text being processed
  - Important for poems or formatted text



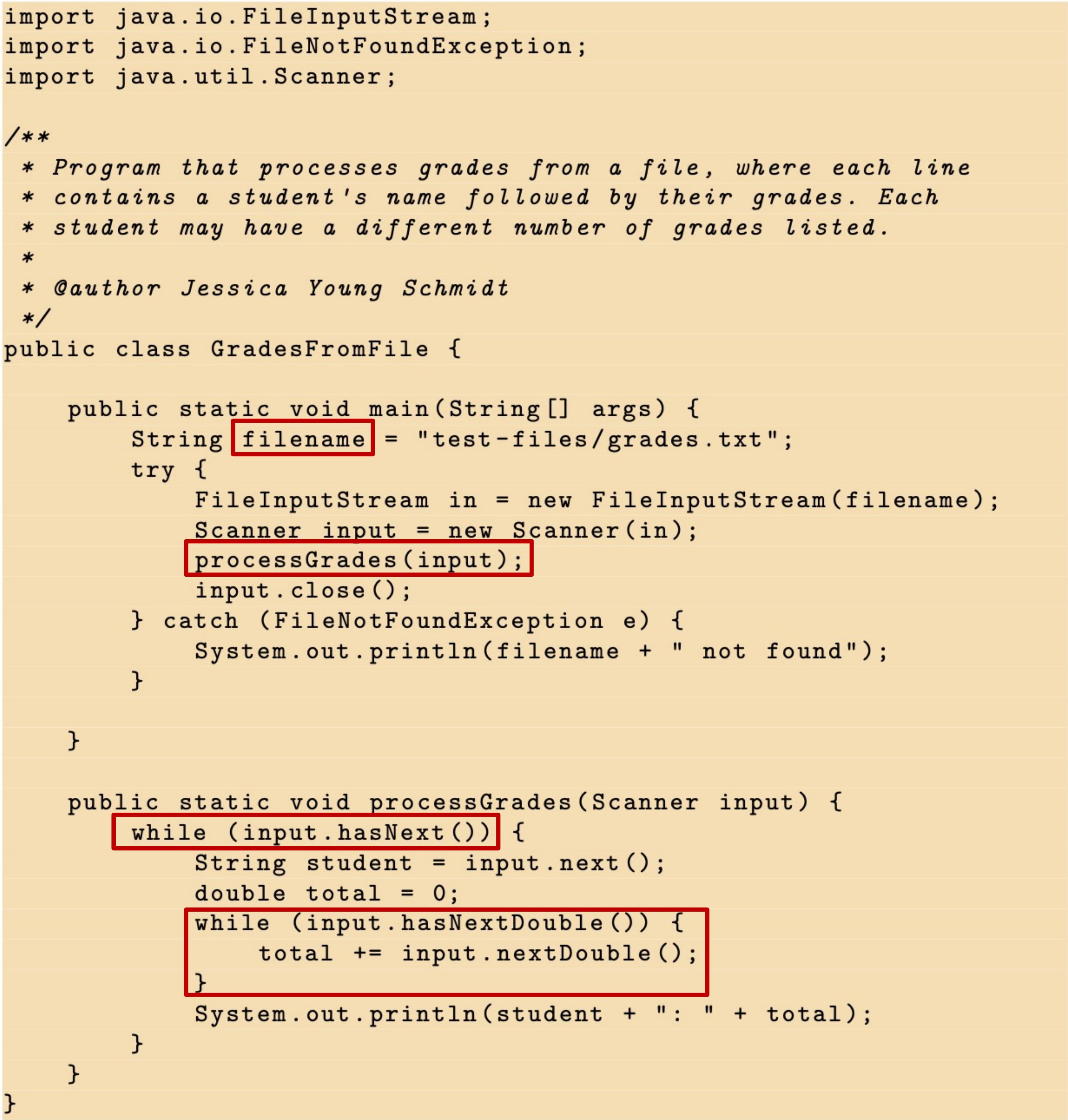
```
Scanner input = new Scanner(new FileInputStream(filename));
while (input.hasNextLine()) {
  String line = input.nextLine();
  Scanner lineScanner = new Scanner(line);
  while (lineScanner.hasNext()) {
    // process each token of line, which may include determining
    // the type of token (e.g., int, double)
    System.out.println(lineScanner.next());
  }
  lineScanner.close();
}
input.close();
```

READING GRADES FROM FILE AND PROCESSING

- First, we are going to examine a token-based approach for an input file in the following format:

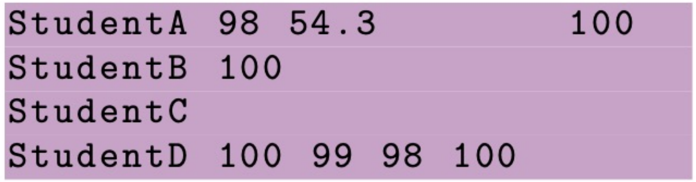


```
StudentA 98 54.3         100
StudentB 100
StudentC
StudentD 100 99 98 100
```

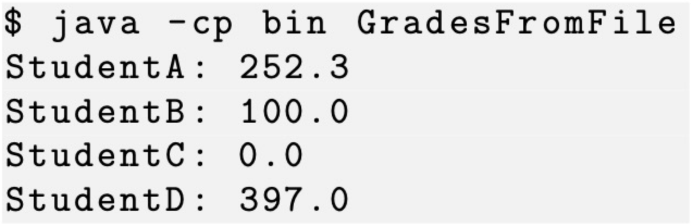


```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.Scanner;

/**
 * Program that processes grades from a file, where each line
 * contains a student's name followed by their grades. Each
 * student may have a different number of grades listed.
 *
 * @author Jessica Young Schmidt
 */
public class GradesFromFile {

    public static void main(String[] args) {
        String filename = "test-files/grades.txt";
        try {
            FileInputStream in = new FileInputStream(filename);
            Scanner input = new Scanner(in);
            processGrades(input);
            input.close();
        } catch (FileNotFoundException e) {
            System.out.println(filename + " not found");
        }

    }

    public static void processGrades(Scanner input) {
        while (input.hasNext()) {
            String student = input.next();
            double total = 0;
            while (input.hasNextDouble()) {
                total += input.nextDouble();
            }
            System.out.println(student + ": " + total);
        }
    }
}
```

Contents of test-files/grades.txt:

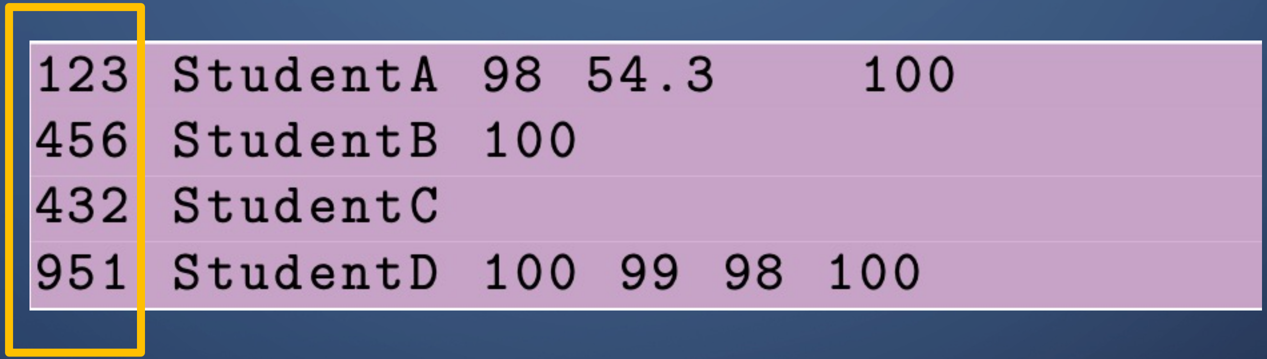


```
StudentA 98 54.3          100
StudentB 100
StudentC
StudentD 100 99 98 100
```



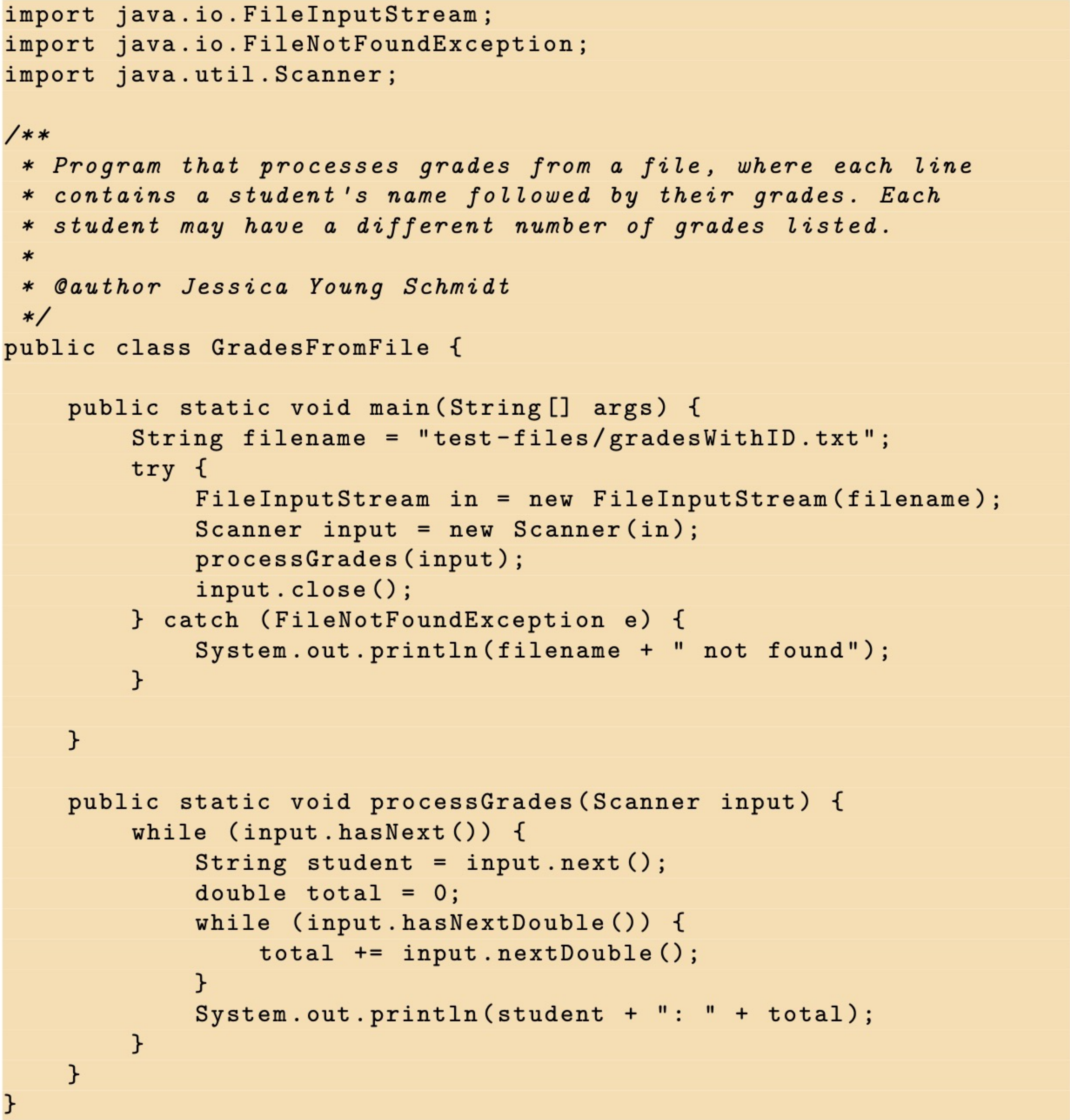
```
$ java -cp bin GradesFromFile
StudentA: 252.3
StudentB: 100.0
StudentC: 0.0
StudentD: 397.0
```

- Next, we are going to examine a line-based approach for an input file in the following format:



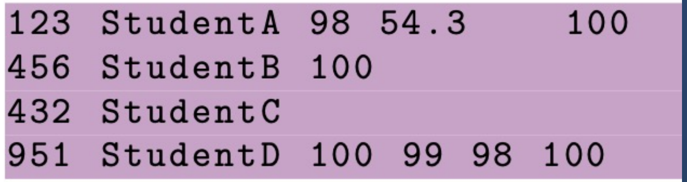
```
123 StudentA 98 54.3    100
456 StudentB 100
432 StudentC
951 StudentD 100 99 98 100
```

- What happens if we continue with a token-based approach?

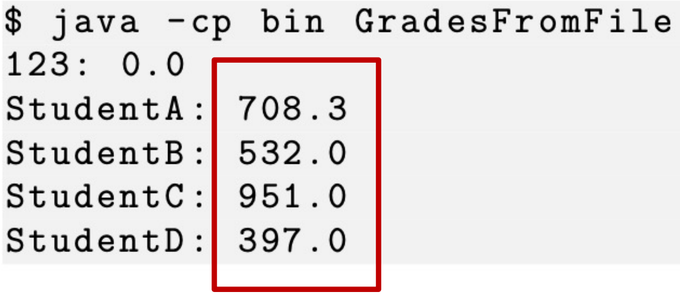


```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.Scanner;

/**
 * Program that processes grades from a file, where each line
 * contains a student's name followed by their grades. Each
 * student may have a different number of grades listed.
 *
 * @author Jessica Young Schmidt
 */
public class GradesFromFile {

    public static void main(String[] args) {
        String filename = "test-files/gradesWithID.txt";
        try {
            FileInputStream in = new FileInputStream(filename);
            Scanner input = new Scanner(in);
            processGrades(input);
            input.close();
        } catch (FileNotFoundException e) {
            System.out.println(filename + " not found");
        }

    }

    public static void processGrades(Scanner input) {
        while (input.hasNext()) {
            String student = input.next();
            double total = 0;
            while (input.hasNextDouble()) {
                total += input.nextDouble();
            }
            System.out.println(student + ": " + total);
        }
    }
}
```

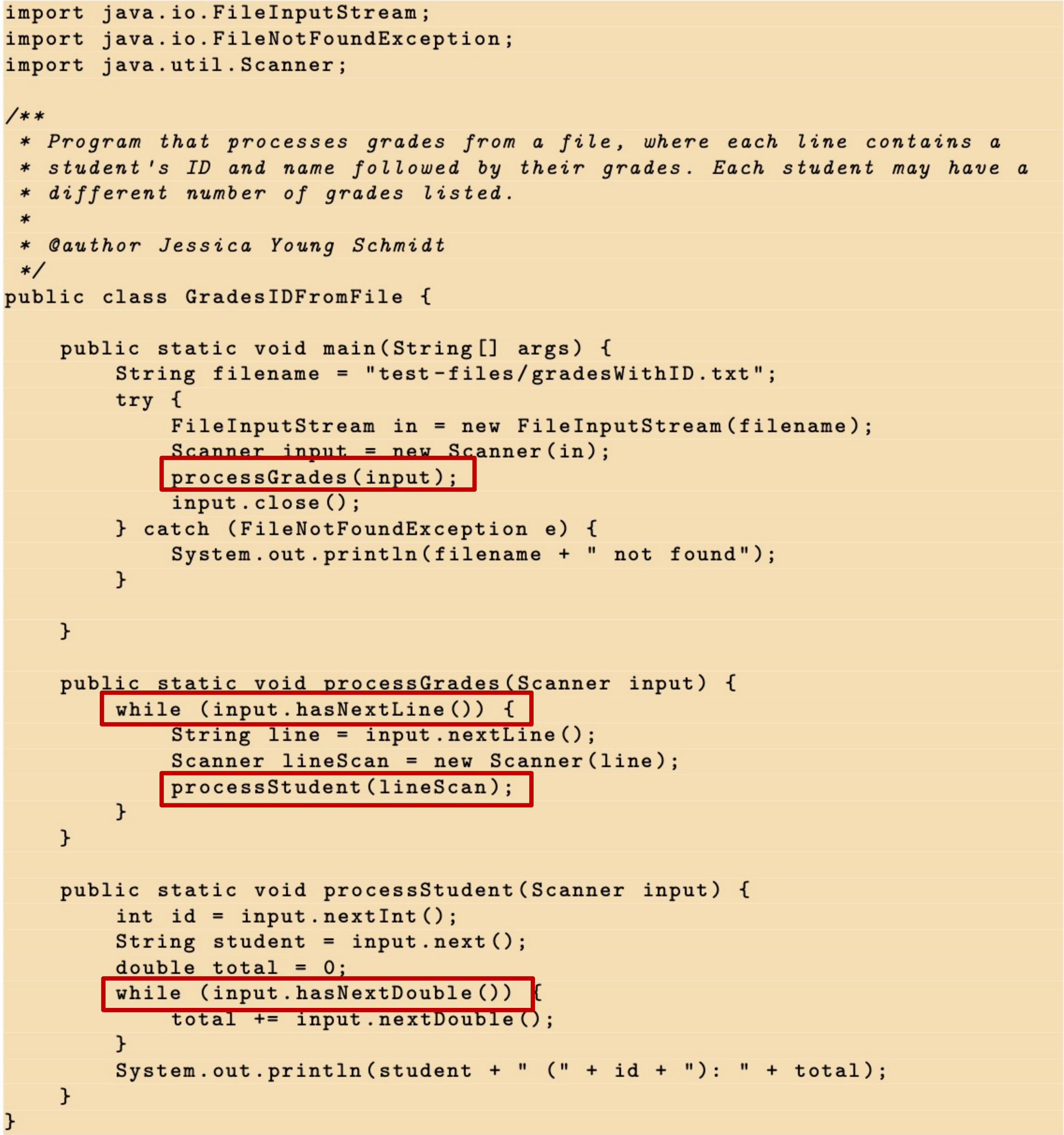
Contents of test-files/gradesWithID.txt:



```
123 StudentA 98 54.3     100
456 StudentB 100
432 StudentC
951 StudentD 100 99 98 100
```

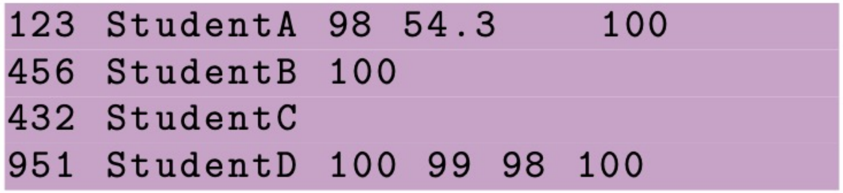


```
$ java -cp bin GradesFromFile
123: 0.0
StudentA: 708.3
StudentB: 532.0
StudentC: 951.0
StudentD: 397.0
```

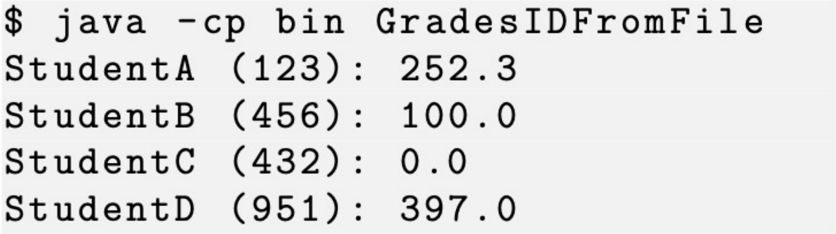


```java
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.Scanner;

/**
 * Program that processes grades from a file, where each line contains a
 * student's ID and name followed by their grades. Each student may have a
 * different number of grades listed.
 *
 * @author Jessica Young Schmidt
 */
public class GradesIDFromFile {

    public static void main(String[] args) {
        String filename = "test-files/gradesWithID.txt";
        try {
            FileInputStream in = new FileInputStream(filename);
            Scanner input = new Scanner(in);
            processGrades(input);
            input.close();
        } catch (FileNotFoundException e) {
            System.out.println(filename + " not found");
        }

    }

    public static void processGrades(Scanner input) {
        while (input.hasNextLine()) {
            String line = input.nextLine();
            Scanner lineScan = new Scanner(line);
            processStudent(lineScan);
        }
    }

    public static void processStudent(Scanner input) {
        int id = input.nextInt();
        String student = input.next();
        double total = 0;
        while (input.hasNextDouble()) {
            total += input.nextDouble();
        }
        System.out.println(student + " (" + id + "): " + total);
    }
}
```

Contents of test-files/gradesWithID.txt:



```
123 StudentA 98 54.3     100
456 StudentB 100
432 StudentC
951 StudentD 100 99 98 100
```



```
$ java -cp bin GradesIDFromFile
StudentA (123): 252.3
StudentB (456): 100.0
StudentC (432): 0.0
StudentD (951): 397.0
```