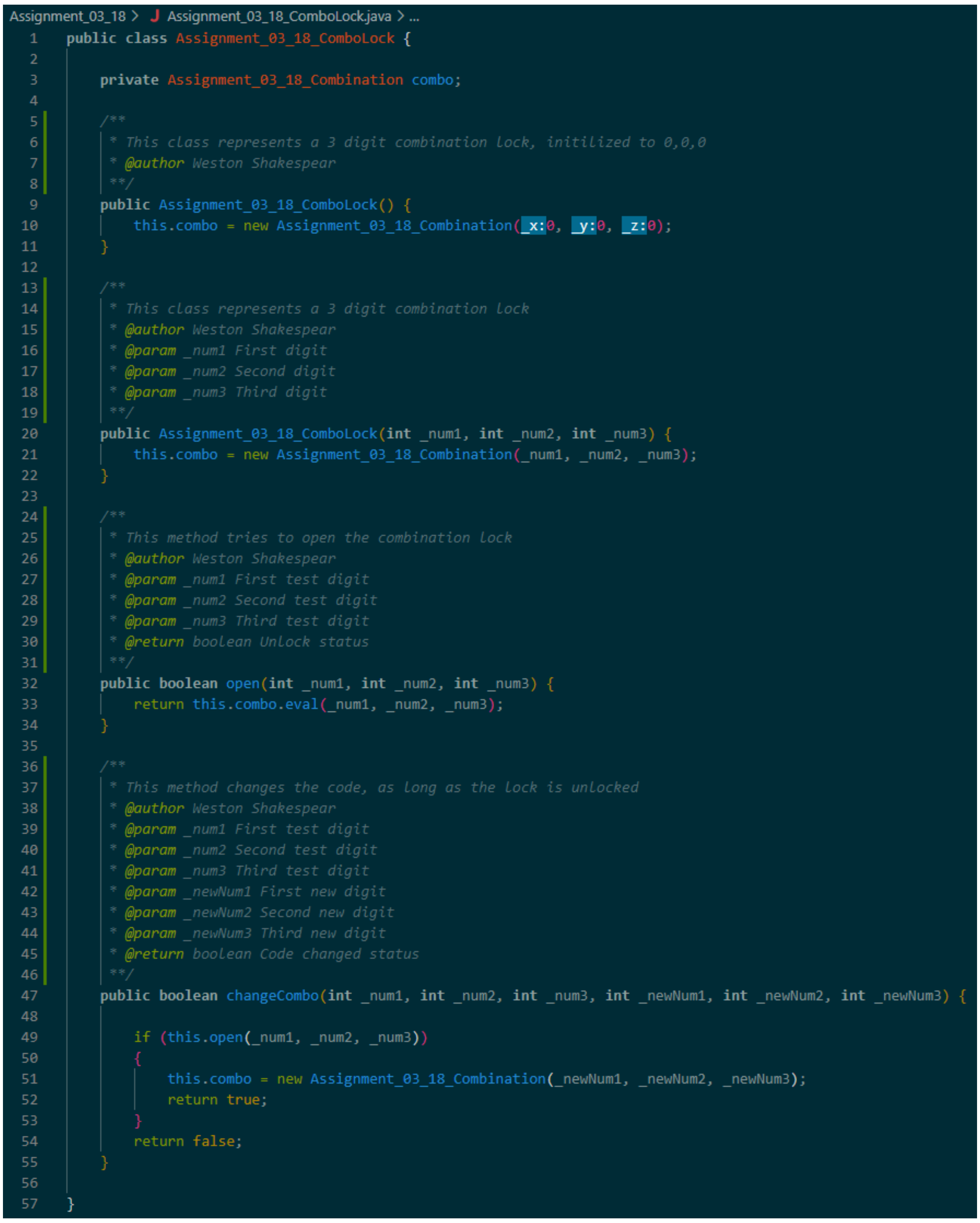
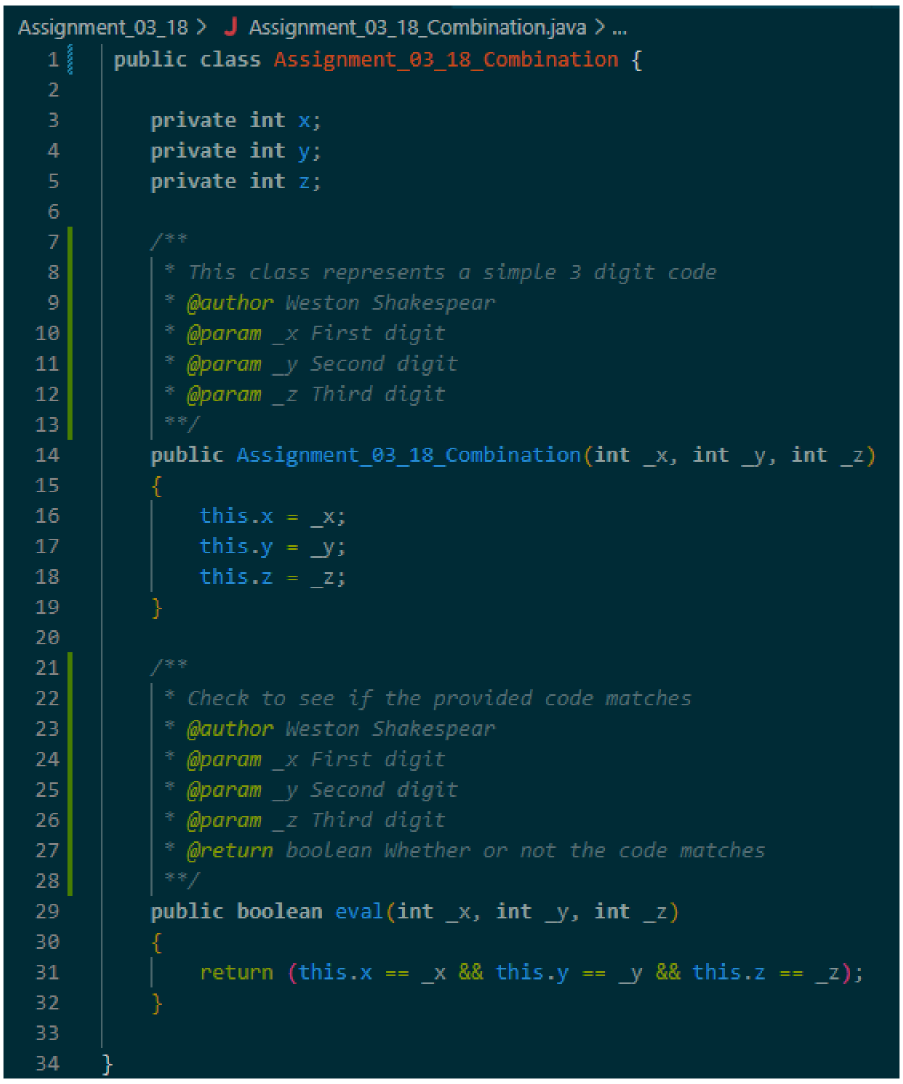
Review: Chapter 3 Programming

3.18 Classes



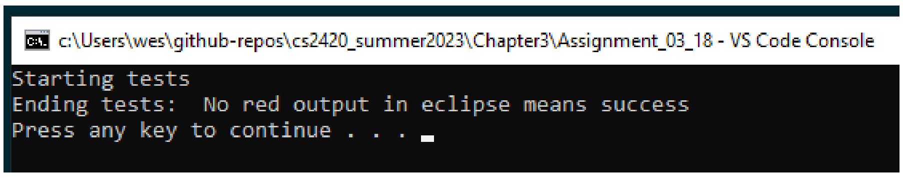
```java
public class Assignment_03_18_ComboLock {

    private Assignment_03_18_Combination combo;

    /**
     * This class represents a 3 digit combination lock, initilized to 0,0,0
     * @author Weston Shakespear
     **/
    public Assignment_03_18_ComboLock() {
        this.combo = new Assignment_03_18_Combination(_x:0, _y:0, _z:0);
    }

    /**
     * This class represents a 3 digit combination lock
     * @author Weston Shakespear
     * @param _num1 First digit
     * @param _num2 Second digit
     * @param _num3 Third digit
     **/
    public Assignment_03_18_ComboLock(int _num1, int _num2, int _num3) {
        this.combo = new Assignment_03_18_Combination(_num1, _num2, _num3);
    }

    /**
     * This method tries to open the combination lock
     * @author Weston Shakespear
     * @param _num1 First test digit
     * @param _num2 Second test digit
     * @param _num3 Third test digit
     * @return boolean Unlock status
     **/
    public boolean open(int _num1, int _num2, int _num3) {
        return this.combo.eval(_num1, _num2, _num3);
    }

    /**
     * This method changes the code, as long as the lock is unlocked
     * @author Weston Shakespear
     * @param _num1 First test digit
     * @param _num2 Second test digit
     * @param _num3 Third test digit
     * @param _newNum1 First new digit
     * @param _newNum2 Second new digit
     * @param _newNum3 Third new digit
     * @return boolean Code changed status
     **/
    public boolean changeCombo(int _num1, int _num2, int _num3, int _newNum1, int _newNum2, int _newNum3) {

        if (this.open(_num1, _num2, _num3))
        {
            this.combo = new Assignment_03_18_Combination(_newNum1, _newNum2, _newNum3);
            return true;
        }
        return false;
    }

}
```



Assignment_03_18 > Assignment_03_18_Combination.java > ...

```java
public class Assignment_03_18_Combination {

    private int x;
    private int y;
    private int z;

    /**
     * This class represents a simple 3 digit code
     * @author Weston Shakespear
     * @param _x First digit
     * @param _y Second digit
     * @param _z Third digit
     **/
    public Assignment_03_18_Combination(int _x, int _y, int _z)
    {
        this.x = _x;
        this.y = _y;
        this.z = _z;
    }

    /**
     * Check to see if the provided code matches
     * @author Weston Shakespear
     * @param _x First digit
     * @param _y Second digit
     * @param _z Third digit
     * @return boolean Whether or not the code matches
     **/
    public boolean eval(int _x, int _y, int _z)
    {
        return (this.x == _x && this.y == _y && this.z == _z);
    }

}
```

3.18 Testing Output

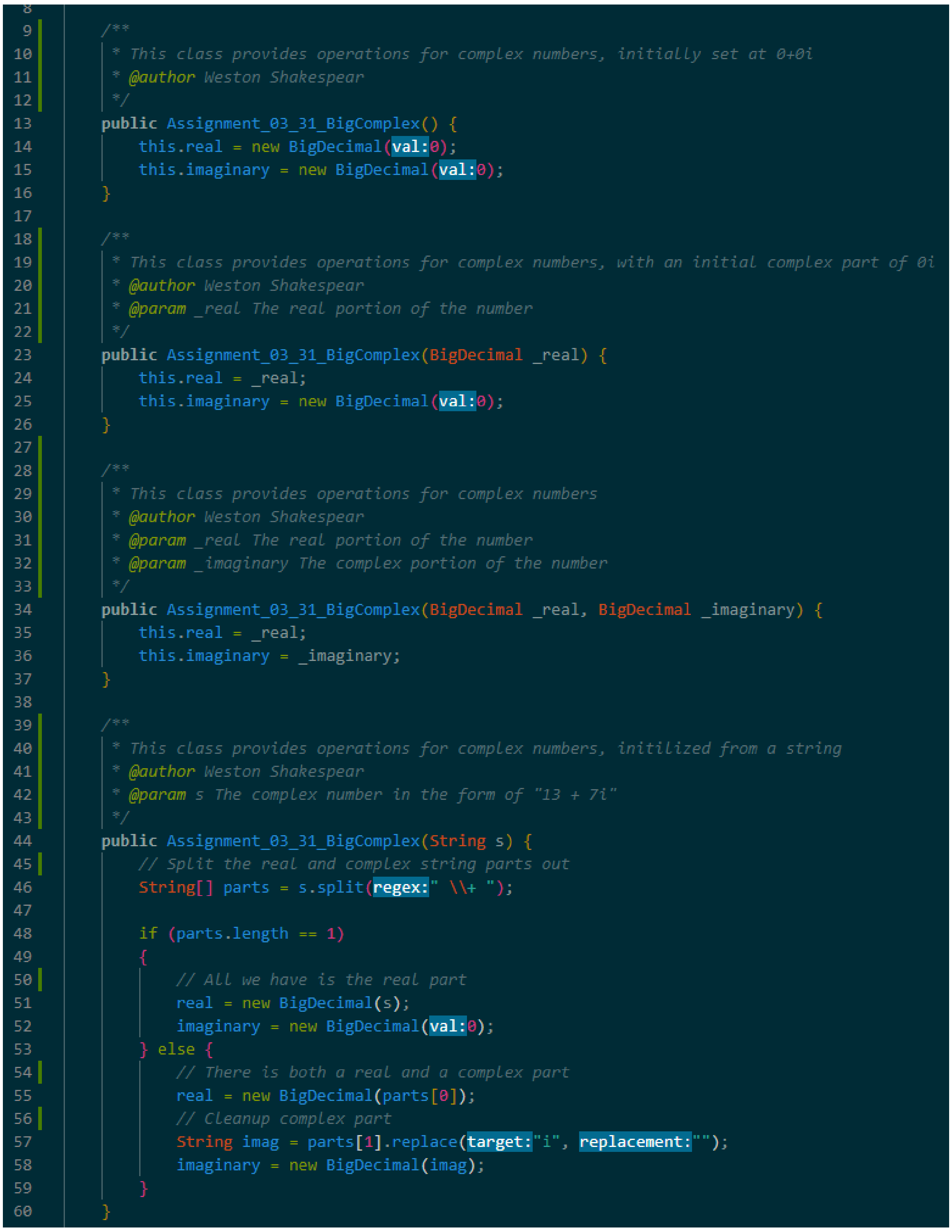


c:\Users\wes\github-repos\cs2420_summer2023\Chapter3\Assignment_03_18 - VS Code Console

```
Starting tests
Ending tests:  No red output in eclipse means success
Press any key to continue . . .
```

Review: Chapter 3 Programming

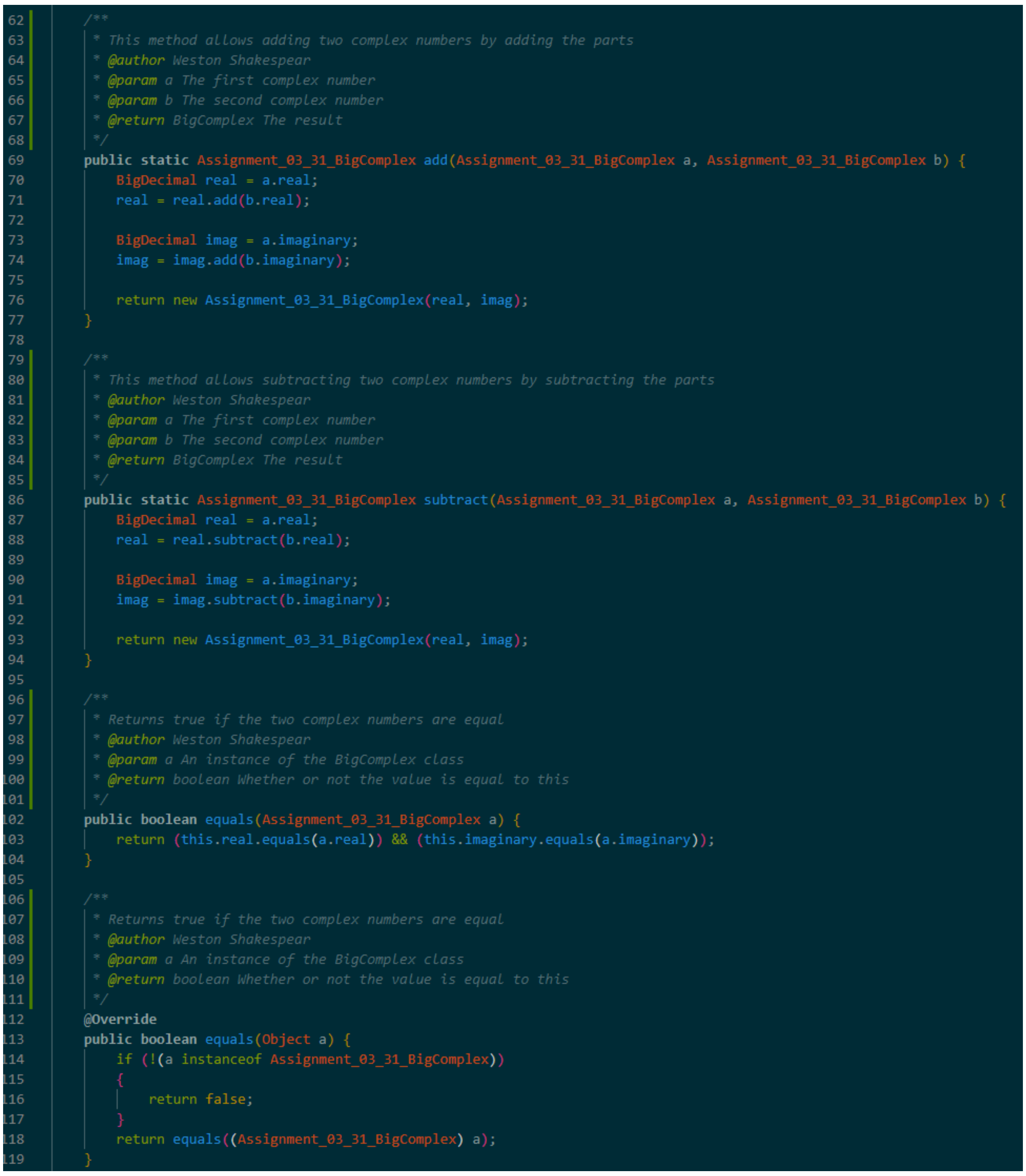
3.31 Constructors



```java
    /**
     * This class provides operations for complex numbers, initially set at 0+0i
     * @author Weston Shakespear
     */
    public Assignment_03_31_BigComplex() {
        this.real = new BigDecimal(val:0);
        this.imaginary = new BigDecimal(val:0);
    }

    /**
     * This class provides operations for complex numbers, with an initial complex part of 0i
     * @author Weston Shakespear
     * @param _real The real portion of the number
     */
    public Assignment_03_31_BigComplex(BigDecimal _real) {
        this.real = _real;
        this.imaginary = new BigDecimal(val:0);
    }

    /**
     * This class provides operations for complex numbers
     * @author Weston Shakespear
     * @param _real The real portion of the number
     * @param _imaginary The complex portion of the number
     */
    public Assignment_03_31_BigComplex(BigDecimal _real, BigDecimal _imaginary) {
        this.real = _real;
        this.imaginary = _imaginary;
    }

    /**
     * This class provides operations for complex numbers, initilized from a string
     * @author Weston Shakespear
     * @param s The complex number in the form of "13 + 7i"
     */
    public Assignment_03_31_BigComplex(String s) {
        // Split the real and complex string parts out
        String[] parts = s.split(regex:" \\+ ");

        if (parts.length == 1)
        {
            // All we have is the real part
            real = new BigDecimal(s);
            imaginary = new BigDecimal(val:0);
        } else {
            // There is both a real and a complex part
            real = new BigDecimal(parts[0]);
            // Cleanup complex part
            String imag = parts[1].replace(target:"i", replacement:"");
            imaginary = new BigDecimal(imag);
        }
    }
```

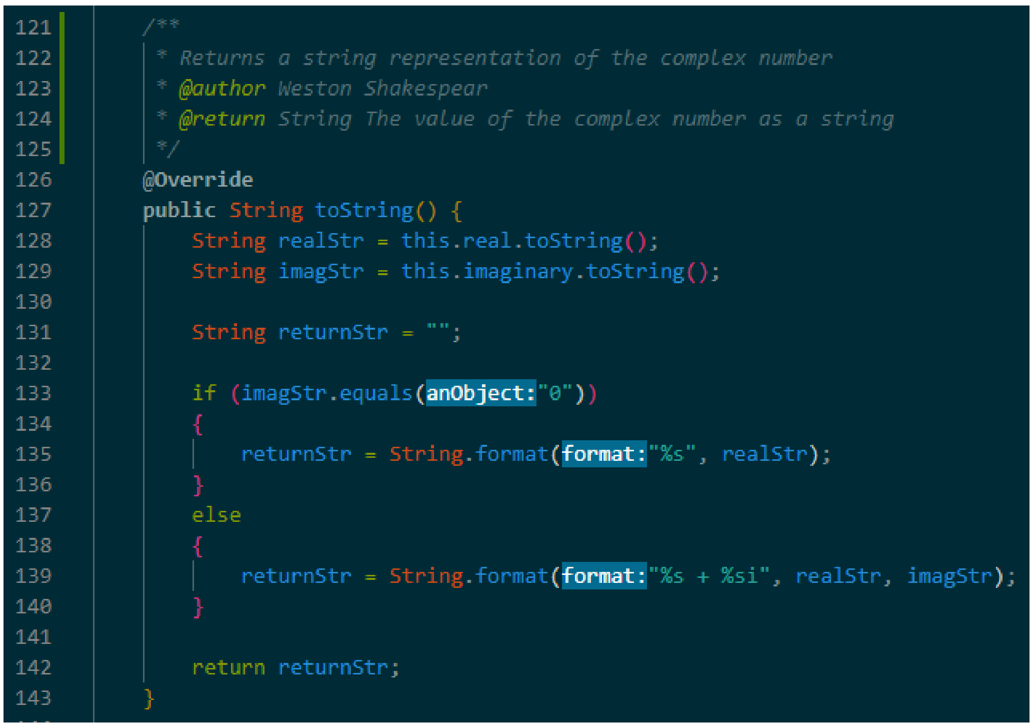
Review: Chapter 3 Programming

3.31 Arithmetic Methods



```java
    /**
     * This method allows adding two complex numbers by adding the parts
     * @author Weston Shakespear
     * @param a The first complex number
     * @param b The second complex number
     * @return BigComplex The result
     */
    public static Assignment_03_31_BigComplex add(Assignment_03_31_BigComplex a, Assignment_03_31_BigComplex b) {
        BigDecimal real = a.real;
        real = real.add(b.real);

        BigDecimal imag = a.imaginary;
        imag = imag.add(b.imaginary);

        return new Assignment_03_31_BigComplex(real, imag);
    }

    /**
     * This method allows subtracting two complex numbers by subtracting the parts
     * @author Weston Shakespear
     * @param a The first complex number
     * @param b The second complex number
     * @return BigComplex The result
     */
    public static Assignment_03_31_BigComplex subtract(Assignment_03_31_BigComplex a, Assignment_03_31_BigComplex b) {
        BigDecimal real = a.real;
        real = real.subtract(b.real);

        BigDecimal imag = a.imaginary;
        imag = imag.subtract(b.imaginary);

        return new Assignment_03_31_BigComplex(real, imag);
    }

    /**
     * Returns true if the two complex numbers are equal
     * @author Weston Shakespear
     * @param a An instance of the BigComplex class
     * @return boolean Whether or not the value is equal to this
     */
    public boolean equals(Assignment_03_31_BigComplex a) {
        return (this.real.equals(a.real)) && (this.imaginary.equals(a.imaginary));
    }

    /**
     * Returns true if the two complex numbers are equal
     * @author Weston Shakespear
     * @param a An instance of the BigComplex class
     * @return boolean Whether or not the value is equal to this
     */
    @Override
    public boolean equals(Object a) {
        if (!(a instanceof Assignment_03_31_BigComplex))
        {
            return false;
        }
        return equals((Assignment_03_31_BigComplex) a);
    }
```

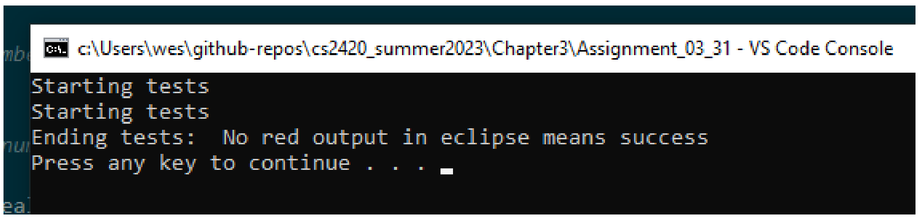
3.31 toString Method



```java
    /**
     * Returns a string representation of the complex number
     * @author Weston Shakespear
     * @return String The value of the complex number as a string
     */
    @Override
    public String toString() {
        String realStr = this.real.toString();
        String imagStr = this.imaginary.toString();

        String returnStr = "";

        if (imagStr.equals(anObject:"0"))
        {
            returnStr = String.format(format:"%s", realStr);
        }
        else
        {
            returnStr = String.format(format:"%s + %si", realStr, imagStr);
        }

        return returnStr;
    }
```

Review: Chapter 3 Programming

3.31 Testing Output



c:\Users\wes\github-repos\cs2420_summer2023\Chapter3\Assignment_03_31 - VS Code Console

```
Starting tests
Starting tests
Ending tests:  No red output in eclipse means success
Press any key to continue . . .
```