# Flyweight Unit Test

```java
package edu.cpp.cs5800.flyweight;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertTrue;

public class DocumentTest {
    @Test
    public void testLoad() {
        Document document = new Document();
        assertTrue(document.load("test_load.txt"));
        String expected = "H(ARIAL,RED,12)\n";
        assertEquals(expected, document.getContent());
    }

    @Test
    public void testAddCharacterOneParam() {
        Document document = new Document();
        Character ch = new Character('Z', new CharacterFormat(CharacterFormat.Font.ARIAL,
                CharacterFormat.Color.BLACK, 14));
        assertTrue(document.addCharacter(ch));

        String expected = "Z(ARIAL,BLACK,14)\n";
        assertEquals(expected, document.getContent());
    }

    @Test
    public void testAddCharacterFourParams() {
        Document document = new Document();
        assertTrue(document.addCharacter('X',
                CharacterFormat.Font.VERDANA,
                CharacterFormat.Color.BLACK,
                14));

        String expected = "X(VERDANA,BLACK,14)\n";
        assertEquals(expected, document.getContent());
    }

    @Test
    public void testSave() {
        String filename = "test_save.txt";
        Character ch = new Character('Z', new CharacterFormat(CharacterFormat.Font.ARIAL,
                CharacterFormat.Color.BLACK, 14));
        Document document = new Document();
        document.addCharacter(ch);
        assertTrue(document.save(filename));

        Document document2 = new Document();
        document2.load(filename);
        assertEquals("Z(ARIAL,BLACK,14)\n", document2.getContent());
    }

    @Test
    public void testGetContent() {
        Character ch1 = new Character('Z', new CharacterFormat(CharacterFormat.Font.ARIAL,
                CharacterFormat.Color.BLACK, 14));
        Character ch2 = new Character('A', new CharacterFormat(CharacterFormat.Font.VERDANA,
                CharacterFormat.Color.BLUE, 16));

        Document document = new Document();
        document.addCharacter(ch1);
        document.addCharacter(ch2);

        String expected = "Z(ARIAL,BLACK,14)\nA(VERDANA,BLUE,16)\n";
        assertEquals(expected, document.getContent());
    }
}

```

# Flyweight Unit Test

```java
package edu.cpp.cs5800.flyweight;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class CharacterTest {
    @Test
    public void testGetColor() {
        CharacterFormat format = new CharacterFormat(CharacterFormat.Font.CALIBRI,
                CharacterFormat.Color.BLACK, 12);
        assertEquals(CharacterFormat.Color.BLACK, format.getColor());
    }

    @Test
    public void testGetFont() {
        CharacterFormat format = new CharacterFormat(CharacterFormat.Font.CALIBRI,
                CharacterFormat.Color.BLACK, 12);
        assertEquals(CharacterFormat.Font.CALIBRI, format.getFont());
    }

    @Test
    public void testGetFontSize() {
        CharacterFormat format = new CharacterFormat(CharacterFormat.Font.CALIBRI,
                CharacterFormat.Color.BLACK, 12);
        assertEquals(12, format.getFontSize());
    }
}

```

# Flyweight Unit Test

```java
package edu.cpp.cs5800.flyweight;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class CharacterFormatTest {
    @Test
    public void testGetSymbol() {
        Character ch = new Character('A', null);
        assertEquals(ch.getSymbol(), 'A');
    }

    @Test
    public void testToString() {
        Character ch = new Character('A', null);
        assertEquals("A(null)", ch.toString());
    }
}

```

# Flyweight Unit Test

```java
package edu.cpp.cs5800.flyweight;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class CharacterFactoryTest {
    @Test
    public void testGetCharacter() {
        String expected = "B(ARIAL,RED,12)";
        Character actual = CharacterFactory.getCharacter('B', CharacterFormat.Font.ARIAL, CharacterFormat.Color.RED, 12);

        assertEquals(expected, actual.toString());
    }
}

```