```
/** test cases for Objectformat operationss **/

import org.junit.Before;
import org.junit.Test;

import junit.framework.TestCase;

public class ObjectFormatTest extends TestCase {
      ObjectFormat stringValue = null;
      String empty = null;
      ArrayFormat value, anotherValue = null;
      ObjectFormat obj, anotherObject = null;

      @Before
      public void setUp() {
            stringValue = ObjectFormat.fromString(
                        "{\"string\":\"Design Patterns\",\"integer
\":1,\"array\":[1,1.2,1800.0,\"string\"],\"double\":198.87}");
            stringValue.put("test", 1000);
            value = new ArrayFormat();
            anotherValue = new ArrayFormat();
            anotherValue.put(12).put("read");
            anotherObject = new ObjectFormat();
            anotherObject.put("key", "value");
            obj = new ObjectFormat();
            value.put(12);
            value.put("cs696");
            value.put(1678.90);
            value.put(anotherObject);
            obj.put("integer", 678);
            obj.put("string", "summer");
            obj.put("double", 189.789);
            obj.put("array", anotherValue);
      }

      @Test
      public void testObjectFormat() throws
ArrayIndexOutOfBoundsException, DataTypeMisMatchException {
            System.out.println("Running testObjectFormat()");
            stringValue.put("score", 78);
            stringValue.put("course", "Assignment3").put("array
value", value).put("object value", obj);
            assertEquals(value, stringValue.getArray("array
value"));
            assertEquals(obj, stringValue.getObject("object
value"));
            assertEquals("Design Patterns",
stringValue.getString("string"));
            assertEquals(198.87, stringValue.getDouble("double"));
            assertEquals((Integer) 1,
stringValue.getInt("integer"));
```

```
          try {
                stringValue.getString("integer");
                fail("Expected an DataTypeMisMatchexception to be
thrown");
          } catch (DataTypeMisMatchException
anDataTypeMisMatchException) {

     assertEquals(anDataTypeMisMatchException.getMessage(),
"value is not of string type");
          }
          try {
                stringValue.put("nothing", empty);
                fail("Expected an IllegalArgumentException to be
thrown");
          } catch (IllegalArgumentException
anIllegalArgumentException) {

     assertEquals(anIllegalArgumentException.getMessage(), "Value
cannot be null");
          }
          assertEquals(9, stringValue.length());
          assertEquals("Assignment3",
stringValue.remove("course"));
          try {
                stringValue.get("illegal");
                fail("Expected an IllegalArgumentException to be
thrown");
          } catch (IllegalArgumentException
anIllegalArgumentException) {

     assertEquals(anIllegalArgumentException.getMessage(), "key
not found");
          }
          assertEquals(4, obj.length());
          obj.put("subject", "history");
          assertEquals(5, obj.length());
          assertEquals(anotherValue, obj.remove("array"));
          assertEquals("{\"key\":\"value\"}",
anotherObject.toString());
     }

}
```