**Assignment Title: Reading XML data in Groovy**

Objective: The objective of this assignment is to provide students with hands-on experience in reading and processing XML data using Groovy. Students will learn how to use Groovy's built-in XML parsing capabilities to read and manipulate data from XML documents.

Instructions:

1. Start by explaining what XML is and why it is used in web development.
2. Introduce Groovy and explain why it is a good language for reading XML data.
3. Provide a sample XML document that contains various elements, attributes, and namespaces.
4. Ask students to write a Groovy script that reads the XML document and extracts specific information, such as element values and attributes.
5. Provide a set of specific requirements for the script, such as the ability to handle nested elements and namespaces.
6. Encourage students to test their scripts with different XML documents to ensure that they are flexible and can handle various scenarios.
7. Ask students to document their code and explain how it works.
8. Provide a set of sample XML documents and ask students to read and process them using their Groovy scripts.
9. Finally, ask students to provide a reflection on what they learned and how they can apply their new knowledge in real-world projects.

Sample XML Document:

<?xml version="1.0" encoding="UTF-8"?>

<catalog>

<book id="bk001">

<author>Writer</author>

<title>The First Book</title>

<genre>Fiction</genre>

<price>44.95</price>

<publish\_date>2000-10-01</publish\_date>

<description>A great book about something or other.</description>

</book>

<book id="bk002">

<author>Writer2</author>

<title>The Second Book</title>

<genre>Non-Fiction</genre>

<price>29.99</price>

<publish\_date>2001-10-01</publish\_date>

<description>Another great book about something or other.</description>

</book>

</catalog>

Requirements:

1. The script should be able to read and parse the sample XML document.
2. The script should be able to extract information from each book element, including element values and attributes.
3. The script should be able to handle nested elements and attributes.
4. The script should be able to output the extracted information in a clear and organized format.
5. The script should be well-documented and easy to understand.

Sample Output:

Book 1:

Author: Writer

Title: The First Book

Genre: Fiction

Price: 44.95

Publish Date: 2000-10-01

Description: A great book about something or other.

Book 2:

Author: Writer2

Title: The Second Book

Genre: Non-Fiction

Price: 29.99

Publish Date: 2001-10-01

Description: Another great book about something or other.